

UNIVERSITY OF BIRMINGHAM



Department of Metallurgy & Materials

Irradiation Damage Simulations of Platinum Group Metal Modified Austenitic Stainless Steels for Reactor Core Components

by Ben Palmer

A thesis submitted to the University of Birmingham for the degree of Doctor of Philosophy
Supervised by Dr Brian J Connolly, Dr Mark S D Read and Prof Alessandro Mottura

Abstract

Austenitic stainless steels have been used since the early days of nuclear power and they will play an important role in the construction of Generation III+ (Gen III+) and Generation IV (Gen IV) plant designs. Materials used in future reactor designs will need to withstand more extreme conditions whilst improving upon safety. Neutron damage is a major contributor to material failure for many complex reasons.

It is known that radiation damage depletes Chromium at the grain boundary. As Chromium is a key element in giving stainless steel its corrosion resistance, a depletion of it leads to inter granular stress corrosion cracking (IGSCC). Corrosion may also be prevented by adding small amounts of Palladium and Ruthenium to the steel, but this may or may not be affected by neutron damage.

Irradiation of components using neutrons is expensive with relatively few high flux reactors available, but even with these reactors it is difficult to speed up the damage process appreciably. A proton beam may be used to emulate neutron damage at a much faster rate and it is also possible to model radiation damage using a computer.

Radioactive waste is created in a reactor when neutrons activate isotopes that make up the components within. This is also true where a light ion beam is used in place of neutrons. The relationship between the ion beam parameters and the component being irradiated is complex and depends upon the type and energy of the projectile, the thickness of the target and its constituent isotopes.

A modified Bateman equation was derived to calculate the radioactivity of a proton irradiated target and a computer program (Activity) was created to implement the calculation using evaluated nuclear reaction cross section data. The program has been compared to data measured from the irradiation of an iron sample using the University of Birmingham cyclotron. It was also used to model the radioactivity of a target irradiated to 100 displacements per atom (DPA) at a range of beam energies in order to minimise both the irradiation time and radioactive waste. The results show an increase in target activity of 3 orders of magnitude for a 25MeV proton beam when compared to a 10MeV proton beam.

Interatomic potentials are required to model radiation damage and its effect using molecular dynamics (MD) or atomic kinetic Monte Carlo (akMC). Platinum group metals PGMs may be added to a stainless steel to improve corrosion resistance, importantly in conditions where chromium is depleted. There are existing potentials for binary alloys of Iron and Palladium as well as for the pure elements Iron, Palladium and Ruthenium, but they are designed for use with particular crystal structures such as body centered cubic (BCC) and hexagonal close packed (HCP).

In this work Density Functional Theory (DFT) is used to create a reference database that is needed for fitting Fe-Pd and Fe-Ru potentials. As the material of interest is austenitic stainless steel, the reference database and potential are restricted to the face centered cubic (FCC) crystal structure. The potential type is a many body embedded-atom method (EAM) that also has a Ziegler-Biersack-Littmark (ZBL) core potential in order to model collisions.

A computer program (EAMPA) was developed in Python and Fortran to fit potentials using the reference database and bulk properties. The resulting potentials are a step towards MD and akMC simulations of radiation damage in PGM doped austenitic stainless steels. They are also a step towards investigating whether or not radiation damage causes PGMs to deplete or enrich at grain boundaries, showing whether or not their corrosion resistance is removed or retained.

Contents

1	Introduction	1
1.1	Motivation	1
1.1.1	Part 1: Activity Computer Program	2
1.1.2	Part 2: Iron-Palladium and Iron-Ruthenium Potential	2
1.2	An Approaching Energy Gap for the UK	2
1.2.1	Why choose Nuclear Power?	2
1.2.2	Proposed Generation III+ Nuclear Power Plant Designs	4
1.2.3	Generation IV	5
1.3	Damage to Nuclear Core Components by Neutrons	10
1.4	Radiation Damage: Replacing Neutrons with Protons	11
1.5	Simulated Material Damage	12
1.5.1	Molecular Dynamics	12
1.5.2	Density Functional Theory	13
1.5.3	Interatomic Potentials for Molecular Dynamics	13
2	Consequences of Ionizing Radiation	15
2.1	Radiation Types Relevant to This Work	15
2.1.1	Introduction	15
2.1.2	Protons and other Ions	15
2.1.3	Neutrons	16
2.1.4	Gamma Rays	17
2.2	Microscopic and Macroscopic Cross Sections	19
2.3	Quantifying Radiation and its Effects	19
2.4	Damage Cascades in Metals	21
3	Austenitic Steels in Nuclear Power	23
3.1	Stainless Steel	23
3.1.1	Introduction	23
3.1.2	Grades of Stainless Steel	23

3.2	Use In Reactors	25
3.2.1	The Use of Austenintic Steels in Reactors	25
3.2.2	Atom Damage Cascades	25
3.2.3	Damage Rates	26
3.2.4	Swelling	28
3.2.5	Radiation Induced Segregation	28
3.3	Corrosion Resistance	30
3.3.1	Passive Film Protection of Chromium	30
3.3.2	Sensitization and Passive Film Removal	30
3.3.3	Addition of Molybdenum	31
3.4	IASCC	31
3.5	IGSCC	31
3.5.1	IGSCC in Light Water Reactors	33
3.5.2	IGSCC and Advanced Gas-cooled Reactors	33
3.6	SS in Gen III+ and Gen IV	34
3.7	Addition of PGMs	34
4	Isotope Activation and Radioactive Decay	37
4.1	Ion Sources	37
4.1.1	LINAC	37
4.1.2	Cyclotron	38
4.1.3	Synchrotron	39
4.2	Neutron Sources	39
4.2.1	High-Flux Neutron Reactors	39
4.2.2	Spallation	40
4.3	Source Review	41
4.4	Emulating Neutron Damage	41
4.4.1	Ion Irradiation at the University of Birmingham	42
4.4.2	Transmutation of Nuclei by Neutrons and Protons	42
4.4.3	Nuclear Reaction Cross Sections	43
4.4.4	Radioactive Decay	47
4.4.5	Saturation Activity	48
4.4.6	Decay Constants: Joint Evaluated Fission and Fusion File (JEFF) 3.3 Data File	49
4.4.7	Bateman Equation for Radioactive Decay	49
4.5	Using SRIM and TRIM	50

5 Interatomic Potential Fitting	53
5.1 Experiment, Modelling and Theory	53
5.1.1 Introduction	53
5.1.2 Simulating Materials on a Variety of Scales in Time and Space	53
5.2 Properties of Crystals	54
5.2.1 Introduction	54
5.2.2 Equation of State	54
5.2.3 Voigt Notation	57
5.2.4 Elastic Constants	57
5.2.5 Calculating Elastic Constants for a Cubic Crystal	58
5.2.6 Calculating Elastic Constants for Orthorhombic Crystal	59
5.2.7 Stability Conditions and Other Properties	61
5.3 Choice of Function	62
5.3.1 Introduction	62
5.3.2 Pair Potentials	62
5.3.3 Many Body Potentials	65
5.3.4 EAM and Two-Band EAM Functions	70
5.4 EAM Potential Fitting	71
5.4.1 Sacrificing Physical Elegance	71
5.4.2 Exact Fitting to Parameters	71
5.4.3 Force Matching	71
5.4.4 Pair and Embedding Symmetry Transforms	72
5.5 Classical Molecular Dynamics	73
5.5.1 Introduction	73
5.5.2 Molecular Dynamics Codes	74
5.5.3 Velocity Verlet Algorithm	76
5.6 Existing Potential Fitting Programs	76
5.7 Density Functional Theory	77
5.7.1 Motivation For Using DFT In This Work	77
5.7.2 Brief Overview of DFT	78
5.7.3 Time Independent Schrödinger Equation	78
5.7.4 Many Body TISE	79
5.7.5 Born-Oppenheimer	80
5.7.6 Crystals, Reciprocal Space and Bloch Theorem	81
5.7.7 Hohenberg-Kohn Theorem	84
5.7.8 Kohn-Sham Equations	86

5.7.9	Self-Consistent Solution	87
5.7.10	Exchange-Correlation Energy	88
5.7.11	Pseudopotentials	89
5.7.12	Ecut, K-Point Integration and Smearing	92
5.7.13	Ferromagnetism, Antiferromagnetism and Hund's Rule	97
5.7.14	Collinear and Non-Collinear DFT Calculations	98
6	Predicting Activation	100
6.1	Introduction	100
6.1.1	Motivation for an Ion Induced Activity Code	100
6.1.2	Summary of Chapter	101
6.2	Activation by Ion Irradiation	101
6.2.1	Laplace Transform	102
6.2.2	Constructing the Differential Equations	102
6.2.3	Numerical Inversion of the Laplace Transform	103
6.2.4	Analytic Solution by Partial Fraction Expansion	104
6.2.5	Preference: Analytic over Numeric	105
6.3	Activity Code	106
6.3.1	Computer Code Development	106
6.3.2	Extended Bateman Equations	106
6.3.3	Talys Generated Proton Cross Section Database	108
6.3.4	Ion Trajectory Data	111
6.3.5	Activity Code Assumptions	112
6.3.6	Foil Activation by Neutron Irradiation	112
6.4	Simulating with SRIM - Iron	112
6.5	Cyclotron Irradiated Iron	112
6.5.1	Cyclotron Beam Line	112
6.5.2	Creation of Cobalt-55	115
6.5.3	Measurement of Sample Activity	116
6.5.4	Prediction of Activity	117
6.5.5	Estimation of Cobalt-55 Radioactivity	118
6.5.6	Simulating Proton Irradiated Iron with Activity V1	119
6.5.7	Simulating Proton Irradiated Iron with Activity V2	120
6.5.8	Comparing Experimental Results with Calculated Values	122
6.6	100DPA Predicted Activity	123
6.7	Proton Irradiated Molybdenum	124

6.7.1	Experimental Irradiation - Molybdenum	125
6.7.2	Estimating the Activity of Proton Irradiated Molybdenum	125
6.7.3	Simulated Proton Irradiated Molybdenum	128
6.8	Proton Irradiated Molybdenum Doped Steel	129
6.9	Media Accessible via GitHub	130
6.10	Concluding Remarks on the Activity Code	130
7	Potentials for Fe, Ru, Pd	131
7.1	Introduction	131
7.1.1	Motivation for Derivation of Potentials	131
7.1.2	Summary of Chapter	132
7.1.3	Interatomic Potential Fitting	132
7.1.4	Existing Experimental & DFT Bulk Property Data	132
7.2	DFT Settings	134
7.2.1	Quantum Espresso	134
7.2.2	Pseudopotential Selection	134
7.2.3	Smearing Type	135
7.2.4	Parameter Convergence	136
7.2.5	QECONVERGE Python Code	137
7.2.6	DFT Parameter Convergence	138
7.3	DFT Results	141
7.3.1	QEEOS Python Program	141
7.3.2	DFT Calculations of Bulk Properties	142
7.3.3	Relaxed Crystal Calculations	142
7.3.4	Elastic Constants	144
7.3.5	Calibrating the Cohesive Energies	148
7.3.6	DFT Reference Database	149
7.3.7	Pure Element Defects	149
7.3.8	Alloy Atom Positions for Pd and Ru	151
7.3.9	Configurations with Randomised Positions	151
7.3.10	Surface Energy Calculations	152
7.3.11	Defects in Fe-Pd and Fe-Ru Alloys	153
7.3.12	DFT Errors, Convergence Failures and Repeat Calculations	153
7.3.13	Summary of Configurations Generated with DFT	153
7.4	EAMPA	153
7.4.1	Introduction	153

7.4.2	Potentials	154
7.4.3	Calculations	154
7.4.4	Verifying Energy and Force Calculations with LAMMPS	158
7.5	Application of the EAMPA Code	158
7.5.1	Potential Transferability	158
7.5.2	Fitting Potentials with EAMPA	160
7.5.3	Iron-Palladium and Iron-Ruthenium Potentials	162
7.6	DL_POLY Contribution	165
8	Conclusions	166
8.1	Inter Granular Stress Corrosion Cracking	166
8.2	Predicting Ion Induced Radioactivity	167
8.2.1	Modified Equation	167
8.2.2	Activity Code	167
8.2.3	Iron Irradiation	168
8.2.4	Molybdenum Irradiation	168
8.2.5	Beam Flux Reduction and Gamma Attenuation	168
8.2.6	More Experimental Data Required	169
8.2.7	100DPA Activity Predictions	169
8.2.8	Original Contribution	169
8.3	Derived Potentials	170
8.3.1	Restricting Potentials to Binary Alloys	170
8.3.2	Collinear Spin DFT	170
8.3.3	Automated Convergence Calculations	170
8.3.4	Bulk Property Calculations	170
8.3.5	Iron-Palladium and Iron-Ruthenium Configurations as Fitting Data	171
8.3.6	Iron-Palladium and Iron-Ruthenium Potentials	171
8.3.7	Original Contribution	172
8.4	Two-Band EAM Contribution to DL-Poly	172
9	Future Work	173
9.1	Activity Code	173
9.1.1	Experimental Activity Readings for Ion Irradiated Targets	173
9.1.2	Code Improvements	173
9.2	Potential Fitting and MD Simulations	174
9.2.1	Larger Supercells	174
9.2.2	Alloy Bulk Properties	174

9.2.3 Defects	175
9.2.4 MEAM for FCC Iron	175
9.2.5 Improved Potential: Fe-Ni-Cr-Pd	175
9.2.6 PKA Cascade Trials in LAMMPS or DL_POLY	175
9.2.7 Radiation Induced Segregation Trials	175
10 Acknowledgements	194
Appendices	195
A Useful Laplace Transforms	196
A.1 Table of Transforms	196
A.2 Decay with a Source Term	196
B Activity Equation in Python: decay class	198
B.1 Full Source Code	198
B.2 Highlighted Code	198
B.3 Activity Equation Testing	208
B.3.1 Numeric Fortran Code	208
B.3.2 Chromium-49	211
B.3.3 Nickel-66	212
B.3.4 Caesium-125	213
B.3.5 Bismuth-213	214
B.3.6 Polonium-216	215
B.3.7 Radon-218	216
C Activity Code	218
C.1 Activity V1	218
C.1.1 Paper published in Computer Physics Communications	218
C.1.2 Accompanying Manual	226
C.2 Activity V2	260
C.2.1 Accompanying Manual	260
D Activity V1 Full Results	285
D.1 36MeV Proton Irradiated Iron	285

E Activity V2 Full Results	297
E.1 36MeV Proton Irradiated Iron	298
E.1.1 Calculation Settings	298
E.1.2 Results File	298
E.1.3 Activity Tallied by Residual Isotope	303
E.1.4 Gamma Output Tallied by Residual Isotope	306
E.1.5 Activity Tallied by Target Isotope	309
E.1.6 Gamma Output Tallied by Target Isotope	312
E.2 13MeV Proton Irradiated Molybdenum	315
E.2.1 Calculation Settings	315
E.2.2 Results	315
E.2.3 Total Decay Rates Over Time	319
E.2.4 Residual Activity Tallied by Target Isotope	320
E.2.5 Residual Activity Tallied by Residual Isotope	321
E.2.6 Residual Gamma Energy Tallied by Residual Isotope	321
E.2.7 Gamma Lines	323
E.3 13MeV Proton Irradiated Steel	324
E.3.1 Calculation Settings	324
E.3.2 Results	324
F Iron Activity: Irradiated to 100DPA	347
F.1 Activity vs Beam Energy	347
F.1.1 5MeV Proton Beam - 60 microamp 100DPA	348
F.1.2 10MeV Proton Beam - 60 microamp 100DPA	350
F.1.3 15MeV Proton Beam - 60 microamp 100DPA	352
F.1.4 20MeV Proton Beam - 60 microamp 100DPA	354
F.1.5 25MeV Proton Beam - 60 microamp 100DPA	356
F.1.6 30MeV Proton Beam - 60 microamp 100DPA	358
G SRIM Results	360
G.0.1 Ion paths through an iron sample	360
H Crystal Structures	362
H.1 Common Cubic Structures	362
H.1.1 Simple Cubic	362
H.1.2 Body Centered Cubic	363
H.1.3 Face Centered Cubic	363

H.1.4 Zincblende	364
H.2 FCC Defects	365
H.2.1 Dumbbell <100>	365
H.2.2 Dumbbell <110>	365
H.2.3 Dumbbell <111>	365
H.2.4 Crowdion	366
H.2.5 Octahedral Interstitial	366
H.2.6 Tetrahedral Interstitial	366
I Elastic Constants	367
I.1 Elastic Constant Matrices	367
I.2 Sets of Strains Applied to Calculate Elastic Constants	369
I.2.1 Cubic Crystals - Mehl, Singh, Klein and Papaconstantopoulos 1993	369
I.2.2 Orthorhombic Crystals - Ravindran, Fast, Korzhavyi and Johansson	370
I.2.3 Cubic Crystals - Connetable and Thomas 2009	371
I.2.4 Orthorhombic Crystals - Connetable and Thomas 2009	371
I.2.5 Stability Conditions	371
I.3 Computing Values from Elastic Constants	372
I.4 Correlation of Melting Temperature in Metals with Elastic Constants	374
J PBE Functional	375
K QECONVERGE	378
L DFT Convergence Plots	380
M QEEOS	387
N DFT Calculated Properties	393
N.1 QEEOS	393
N.2 FCC Aluminium	394
N.2.1 DFT Settings for PWscf	394
N.2.2 Resulting Plots	394
N.2.3 Aluminium FCC Results File	395
N.2.4 Known and Calculated Values	396
N.3 BCC Iron [No Magnetism]	397
N.3.1 Major DFT Settings	397
N.3.2 Equation of State and Elastic Constants	397
N.3.3 Iron BCC Results File	398

N.4	BCC Iron [Ferromagnetic]	401
N.4.1	Major DFT Settings	401
N.4.2	Equation of State and Elastic Constants	401
N.4.3	Iron Ferromagnetic BCC Results File	402
N.4.4	Known and Calculated Results	405
N.5	FCC Iron [Antiferromagnetic]	406
N.5.1	Major DFT Settings	406
N.5.2	Equation of State and Elastic Constants	406
N.5.3	Iron FCC Results File	407
N.6	FCC Palladium	410
N.6.1	Major DFT Settings	410
N.6.2	Equation of State and Elastic Constants	410
N.6.3	Palladium FCC Results File	411
N.6.4	Known and Calculated Values	413
N.7	FCC Ruthenium	414
N.7.1	Major DFT Settings	414
N.7.2	Equation of State and Elastic Constants	414
N.7.3	Ruthenium FCC Results File	415
N.7.4	Calculated Results	417
O	DFT Errors	418
O.0.1	Insufficient Memory	418
O.0.2	Scratch Restrictions	418
O.0.3	Caching Repeat Calculations	419
O.0.4	Adjustment of DFT Parameters	419
O.0.5	Not Straying Too Far From Reality	420
O.0.6	Trial Parameters for Iron with an Octahedral Defect	420
P	Transferability	422
P.1	Sheng Aluminium Potential	422
P.1.1	FCC Structure	422
P.1.2	BCC Structure	424
P.2	Mendelev et al 2003 Iron Potential	424
P.2.1	BCC Structure	424
P.2.2	FCC Structure	424
P.3	Ackland et al 1997 Iron Potential	424
P.3.1	BCC Structure	424
P.3.2	FCC Structure	424

Q Optimization	430
Q.1 Optimization	430
Q.1.1 Introduction	430
Q.1.2 Continuous and Discrete Optimization	430
Q.1.3 Constrained and Unconstrained Optimization	430
Q.1.4 Global and Local Optimization	431
Q.1.5 Global Optimization Algorithms	431
Q.1.6 Local Optimization Algorithms	433
R EFS Calculations	436
R.0.1 Calculating Energy, Force and Stress	436
S EAMPA Code	442
S.1 Energy, Force, and Stress Subroutine	442
S.2 Neighbour List Subroutine	445
S.3 Fortran Programmed Functions	447
S.4 Main fitting class	450
S.5 Global Fitting Algorithms	456
S.6 Bulk Properties	462
S.7 Interpolation	474
S.8 Interpolation (Gradient)	474
T Potential Functions	476
T.1 Introduction	476
T.2 Functions	476
T.2.1 Ackland Embedding	476
T.2.2 Buckingham	477
T.2.3 Lennard Jones	478
T.2.4 Mishin Density	479
T.2.5 Morse	481
T.2.6 Polynomial Embedding	482
T.2.7 Spline	483
T.2.8 Slater 4s	486
T.2.9 Zero	488
T.3 Splines	489
U Fe-Pd and Fe-Ru Structures	492

V Fe-Pd Potential	495
V.1 Tabulated Potential	495
V.2 Potential Function Plots	496
V.3 Potential vs DFT	497
V.4 Bulk Properties	498
V.4.1 Iron FCC	498
V.4.2 Palladium FCC	500
V.4.3 Iron BCC	502
W Fe-Ru Potential	504
W.1 Tabulated Potential	504
W.2 Potential Function Plots	505
W.3 Potential vs DFT	506
W.4 Bulk Properties	507
W.4.1 Iron FCC	507
W.4.2 Ruthenium FCC	509
W.4.3 Iron BCC	511
X Contribution to the DLPOLY Source Code	513
X.1 Contribution to DLPOLY 4.04	513

List of Figures

1.1	Electricity in Millions of Tonnes of Oil Equivalent[2]	3
1.2	Safest and cleanest sources of energy [5] - deaths per terawatt hour and tonnes of CO ₂ per gigawatt hour	4
1.3	Typical reactor flux - thermal and fast spectra [12]	6
1.4	Inelastic scattering - scattering angle vs energy loss for a range of target atoms - generated with a python script using the equations contained in Duderstadt and Hamilton: Nuclear Reactor Analysis[14]	7
1.5	Energy demand to produce hydrogen by electrolysis as temperature changes[18]	8
1.6	Phases of water: pressure vs temperature [23]	9
1.7	Neutron energy spectrum from fission generated from the JEFF 3.1.1 data file[26]	10
1.8	PKA energy as a function of neutron energy in Fe[27]	11
1.9	Damage cascades in Fe for a range of energies generated in SRIM[32]	13
2.1	Proton (p) and neutron (n) interactions with the exchange of virtual pi mesons (Π^0 , Π^- , Π^+), carriers of the strong nuclear force, responsible for binding neutrons or protons to the nucleus during the activation of a material	16
2.2	Pair production Feynman diagram, the process responsible for 511keV and 1.022MeV gamma lines[35]	18
2.3	Compton Scattering Feynman diagram - responsible for attenuation of gamma rays through a material[36]	18
2.4	Photoelectric Effect Feynman diagram - absorption of gamma rays through the transfer of energy to an orbital electron[37]	18
2.5	Interstitial formation in 10keV, 20keV and 40keV PKA displacement cascades and subcascades [48]. Only interstitials resulting from the three PKAs are shown in the simulation cell.	22
3.1	Steel Cr-Ni Schaeffler Diagram[49]	24
3.2	Iron-Chromium-Nickel Phase Diagram[50]	24
3.3	BCC structure of ferritic stainless steel - a 2x2x2 supercell[51]	24
3.4	FCC structure of austenitic stainless steel - a 2x2x2 supercell[51]	24
3.5	Expected DPA during component life time and operating temperatures[54]	26
3.6	Scattering angle probability of neutrons plotted with a self written Python script using eq. 3.1[40][14] (elastic scattering only)	27

3.7	Energy transfer as a from a neutron dependant on target mass plotted with a self written Python script using eq. 3.1[40][14] with the scattering angle set to π radians, i.e. scattering back and transferring maximum energy to the target nucleus (elastic scattering only)	27
3.8	Recoil energies for neutrons at 102keV, 1.02MeV, 10.1MeV and 14.1MeV[27]	27
3.9	Depletion of cr, fe and enrichment of ni at the surface[64]	29
3.10	Concentration profile after 46DPA irradiation with 75 keV Ni ions[65]	29
3.11	Temperature and DPA dependence of RIS[66]	29
3.12	Formation of chromium precipitates at the grain boundary [71]	30
3.13	Sensitization of an alloy: chromium carbide precipitation [72]	30
3.14	Changing water chemistry: radiolysis of water[77]	31
3.15	Inter Granular Stress Corrosion Cracking in Nickel Alloy[78]	32
3.16	Three requirements for IGSCC to occur	32
3.17	Alloy choices for early LWRs[79]	33
3.18	Nickel, IGSCC and TGSCC[80]	33
3.19	Addition of Ru to Fe30Cr and Fe40Cr steels - open current corrosion potential variation with time in 10% sulphuric acid[88]	35
3.20	Failure times for 304SS, Pd doped 304SS (with and without Mn) and Ru doped 304SS in polythionic acid at ambient temperature[89]	35
4.1	A plot of relativistic and classical velocities for 0MeV - 1000MeV protons	38
4.2	High Flux Isotope Reactor	39
4.3	NBSR building layout[97]	40
4.4	An example of a neutron spallation source where high energy ions collide with heavy atoms[99]	40
4.5	Neutron Spectra from the Fission of U235[101]	41
4.6	Neutron, proton and deuteron cross sections for Fe56 and Ni58, plotted with data from the TENDL database[104]	42
4.7	TALYS work flow [114]	45
4.8	TALYS Fe54-Co55 cross section comparison with experiment [115]	46
4.9	An example decay chain from an unstable parent isotope, through unstable daughter isotopes ending with a stable daughter isotope.	49
4.10	One hundred simulated 13MeV proton energy loss curves in Fe simulated with SRIM [32]	52
5.1	Time and Size Scales for Computer Packages [119]	54
5.2	Rose-Vinet Equation of State[125]	57
5.3	Lennard-Jones Potential	62
5.4	Morse Potential	63
5.5	Buckingham Potential	64
5.6	ZBL universal screening potential	65
5.7	Damage in MOX (LAMMPS)[147]	74

5.8	Sputtering of plutonium oxide (LAMMPS)[148]	74
5.9	5keV cascade (MOLDY)[149]	75
5.10	20keV cascade (MOLDY)[149]	75
5.11	100keV cascade after 50ps in TiO ₂ (DL_POLY)[150]	75
5.12	Common approximations used to enable DFT calculations	78
5.13	A useful, albeit incorrect, way of visualising an "infinite" chain of atoms in 1D	82
5.14	A 2D example of a translation by an integer multiple of the unit cell from the origin unit cell	82
5.15	BCC Wigner Seitz Cell and BZ [158]	83
5.16	FCC Wigner Seitz Cell and BZ [158]	83
5.17	SCF algorithm for Quantum Espresso	87
5.18	Replacing the complex potentials with a pseudopotential[170]	89
5.19	Successes and failures of the GGA functional[168]	92
5.20	Energy gap between valence and conduction bands[181]	94
5.21	Probability F(E) of finding a fermion with energy E at several temperatures[182]	95
5.22	Delta function replaced by successive Hermite polynomials[183]	96
5.23	The occupancy for the Marzari-Vanderbilt smearing function is always positive, unlike the Methfessel-Paxton function[184]	96
6.1	An example of several decay chains including branching factors and possible external source terms for each isotope on each chain.	102
6.2	Decay of Po-218: Analytic and Gaver-Stehfest Calculations [26]	106
6.3	Decay path of Polonium-216	107
6.4	Decay path of Polonium-216	108
6.5	Data from pre-existing TENDL files for (⁵⁶ Fe, p) reactions	109
6.6	Data from TALYS generated files for (⁵⁶ Fe, p) reactions	110
6.7	36MeV proton tracks into a 0.5mm thick steel generated with the SRIM code where protons enter the material from the left of the figure and exit on the right.	111
6.8	Protons through a 0.5mm thick iron target - 100 proton histories	113
6.9	Protons through target thick enough to stop all ions - 100 proton histories	114
6.10	Cyclotron layout - main hall and beam lines	114
6.11	Gamma warning alarm during 0.5 microamp 36MeV proton irradiation of iron	114
6.12	Proton stopping distances in various materials	115
6.13	Decay path of Cobalt-55	115
6.14	Cobalt-55 Gamma Intensity Plot	116
6.15	High Purity Germanium Detector	116
6.16	Gamma counts from the irradiated iron target over 300 seconds	117
6.17	<i>Fe</i> 56(<i>p</i> , 2 <i>n</i>) <i>Co</i> 55 cross section data from several sources [104][196][197][198][199]	118

6.18 Isotopes with the top 5 radioactivity plotted over approximately 3 days	120
6.19 Predicted gamma lines approximately 3 days after irradiation	120
6.20 Particle production rate in beam	121
6.21 Gamma output of residual isotopes, by isotope, at the start and end of irradiation	121
6.22 Residual isotope power output, by residual, once beam is off.	122
6.23 Expected gamma lines by the end of the experiment (3 days)	122
6.24 Change in isotope gamma output over time with proton beam off.	124
6.25 Proton irradiated Molybdenum gamma activity measured by colleague John Hewett	125
6.26 Isotopes in naturally occurring Molybdenum	126
6.27 Reaction cross sections for $^{96}\text{Mo}(p,n)^{96}\text{Tc}$ from various sources	127
6.28 Tc activity and beam energy as a function of depth into the target, showing a non-uniform distribution of radioactive atoms in the sample	128
6.29 The attenuation of Gammas through Mo for a range of energies and thicknesses[39]	128
6.30 Comparison of Gamma peaks from experiment, a simple estimate and the Activity code.	129
7.1 Work flow used to fit potentials	133
7.2 Equation of state fit through data points for Chromium BCC with no magnetism, ferromagnetic and antiferromagnetic configurations	139
7.3 Equation of state fit through data points for Iron BCC (no magnetism, ferromagnetic) and Iron FCC (no magnetism and antiferromagnetic)	140
7.4 Magnetic alignment along the z-axis, antiferromagnetic Fe FCC	144
7.5 Isolated atom energies as cell size increase	148
7.6 Neighbouring atom separations and their frequency over 5 subsections of the DFT reference database. The neighbour separation, in angstrom, runs along the x-axis. The neighbour separations were sorted into 200 bins for each reference database, and the percentage each bin is filled runs along the y-axis.	150
7.7 Binary alloy configurations with less than 2% PGMs	151
7.8 Effect of randomly perturbing atom locations	152
7.9 A sample of configurations used for binary alloy DFT calculations for Fe-Pd and Fe-Ru.	152
7.10 Parameter Breed Event	157
7.11 DIRAC calculated radial electron density for Iron	160
7.12 DIRAC calculated radial electron density for Palladium	160
7.13 Rose-Vinet plots for Iron	163
7.14 Rose-Vinet plots for Palladium and Ruthenium	163
7.15 Plotting how well the potentials reproduce DFT energy values	164
7.16 Energy plotted as both a slab and two surfaces begin to form. The plots give the energies predicted by DFT and computed using the potentials from this work.	164
B.1 Decay path for Chromium-49 [26]	211

B.2 Decay path for Nickel-66 [26]	212
B.3 Decay path for Caesium-125 [26]	213
B.4 Decay path for Bismuth-213 [26]	214
B.5 Decay path for Bismuth-213 [26]	215
B.6 Decay path for Bismuth-213 [26]	216
E.1 Residual isotope activity by isotope at 3 seconds into irradiation	303
E.2 Residual isotope activity by isotope at 60 seconds into irradiation	303
E.3 Residual isotope activity by isotope at 300 seconds into irradiation - end of irradiation	304
E.4 Residual isotope activity by isotope at 1 day, with approximately 1 day of cooling	304
E.5 Residual isotope activity by isotope at 2 days, with approximately 2 days of cooling	305
E.6 Residual isotope activity by isotope at approximately 3 days, with approximately 3 days of cooling	305
E.7 Gamma activity by isotope at 3 seconds into irradiation	306
E.8 Gamma activity by isotope at 60 seconds into irradiation	306
E.9 Gamma activity by isotope at 300 seconds into irradiation	307
E.10 Gamma activity by isotope with approximately 1 day of cooling	307
E.11 Gamma activity by isotope with approximately 2 days of cooling	308
E.12 Gamma activity by isotope with approximately 3 days of cooling	308
E.13 Isotope activity contribution by target isotope at 3 seconds into irradiation	309
E.14 Isotope activity contribution by target isotope at 60 seconds into irradiation	309
E.15 Isotope activity contribution by target isotope at 300 seconds into irradiation	310
E.16 Isotope activity contribution by target isotope with approximately 1 day of cooling	310
E.17 Isotope activity contribution by target isotope with approximately 2 days of cooling	311
E.18 Isotope activity contribution by target isotope with approximately 3 days of cooling	311
E.19 Gamma activity contribution by target isotope at 3 seconds into irradiation	312
E.20 Gamma activity contribution by target isotope at 60 seconds into irradiation	312
E.21 Gamma activity contribution by target isotope at 300 seconds into irradiation	313
E.22 Gamma activity contribution by target isotope with approximately 1 day of cooling	313
E.23 Gamma activity contribution by target isotope with approximately 2 days of cooling	314
E.24 Gamma activity contribution by target isotope with approximately 3 days of cooling	314
E.25 Total activity over time	319
E.26 Total gamma power output over time	319
E.27 Radioactive decay from residual isotopes tallied by target isotope - end of beam (1,500 seconds) .	320
E.28 Radioactive decay from residual isotopes tallied by target isotope - 3 days after irradiation . . .	320
E.29 Radioactive decay from residual isotopes tallied by residual isotope - end of beam (1,500 seconds)	321
E.30 Radioactive decay from residual isotopes tallied by residual isotope - 3 days after irradiation . .	321

E.31 Radioactive decay from residual isotopes tallied by residual isotope - end of beam (1,500 seconds)	322
E.32 Radioactive decay from residual isotopes tallied by residual isotope - 3 days after irradiation	322
E.33 Predicted gamma lines - end of beam (1,500 seconds)	323
E.34 Predicted gamma lines - 3 days after irradiation	323
F.1 Cooling measured by gamma power for iron samples irradiated to 100DPA at a range of proton energies	347
F.2 Residual isotope activity over time with 5MeV protons	348
F.3 Residual isotope gamma output at the end of irradiation with 5MeV protons	348
F.4 Residual isotope gamma output after 1 week of cooling	349
F.5 Residual isotope activity over time with 10MeV protons	350
F.6 Residual isotope gamma output at the end of irradiation with 10MeV protons	350
F.7 Residual isotope gamma output after 1 week of cooling	351
F.8 Residual isotope activity over time with 15MeV protons	352
F.9 Residual isotope gamma output at the end of irradiation with 15MeV protons	352
F.10 Residual isotope gamma output after 1 week of cooling	353
F.11 Residual isotope activity over time with 20MeV protons	354
F.12 Residual isotope gamma output at the end of irradiation with 20MeV protons	354
F.13 Residual isotope gamma output after 1 week of cooling	355
F.14 Residual isotope activity over time with 25MeV protons	356
F.15 Residual isotope gamma output at the end of irradiation with 25MeV protons	356
F.16 Residual isotope gamma output after 1 week of cooling	357
F.17 Residual isotope activity over time with 30MeV protons	358
F.18 Residual isotope gamma output at the end of irradiation with 30MeV protons	358
F.19 Residual isotope gamma output after 1 week of cooling	359
G.1 SRIM trajectories for 12MeV, 14MeV and 16MeV protons in iron	360
G.2 SRIM trajectories for 18MeV, 20MeV and 22MeV protons in iron	360
G.3 SRIM trajectories for 24MeV, 26MeV and 28MeV protons in iron	360
G.4 SRIM trajectories for 30MeV, 32MeV and 34MeV protons in iron	361
G.5 SRIM trajectories for 36MeV protons in iron	361
I.1 Correlation between C_{11} and temperature	374
L.1 Ecutfc and Ecutrho convergence for aluminium (energy and force)	381
L.2 2D density map - force and energy convergence with k-points and smearing - alluminium	382
L.3 Ecutfc and Ecutrho convergence for iron (energy and force)	383
L.4 2D density map - force and energy convergence with k-points and smearing - iron	384

L.5	Ecutwfc and Ecutrho convergence for palladium (energy and force)	385
L.6	2D density map - force and energy convergence with k-points and smearing - palladium	386
N.1	Aluminium FCC equation of state plot	394
N.2	Aluminium FCC elastic constants plot	394
N.3	Iron BCC equation of state plot	397
N.4	Iron BCC elastic constants plot	397
N.5	Iron BCC (magnetic) equation of state plot	401
N.6	Iron BCC (magnetic) elastic constants plot	401
N.7	Iron FCC (magnetic) equation of state plot	406
N.8	Iron FCC (magnetic) elastic constants plot	406
N.9	Palladium FCC equation of state plot	410
N.10	Palladium FCC elastic constants plot	410
O.1	Fe-Ru calculations failed to converge in 100 iterations when plain mixing was used, but achieved convergence in under 40 iterations with local-TF convergence	420
P.1	Birch Murnaghan equation of state for FCC Aluminium - comparison of experimental data to the Sheng Al potential	423
P.2	Rose-Vinet equation of state for FCC Aluminium - comparison of experimental data to the Sheng Al potential	423
P.3	Birch Murnaghan equation of state for BCC Aluminium - comparison of experimental data to the Sheng Al potential	425
P.4	Rose-Vinet equation of state for BCC Aluminium - comparison of experimental data to the Sheng Al potential	425
P.5	Birch Murnaghan equation of state for FCC Aluminium - comparison of experimental data to the Sheng Al potential	426
P.6	Rose-Vinet equation of state for FCC Aluminium - comparison of experimental data to the Sheng Al potential	426
P.7	Birch Murnaghan equation of state for FCC Aluminium - comparison of experimental data to the Sheng Al potential	427
P.8	Rose-Vinet equation of state for FCC Aluminium - comparison of experimental data to the Sheng Al potential	427
P.9	Birch Murnaghan equation of state for FCC Aluminium - comparison of experimental data to the Sheng Al potential	428
P.10	Rose-Vinet equation of state for FCC Aluminium - comparison of experimental data to the Sheng Al potential	428
P.11	Birch Murnaghan equation of state for FCC Aluminium - comparison of experimental data to the Sheng Al potential	429
P.12	Rose-Vinet equation of state for FCC Aluminium - comparison of experimental data to the Sheng Al potential	429

R.1	Neighbouring atom count dependant upon lattice parameter and cutoff radius	437
R.2	A 2D representation of atoms in simulation box with a halo of atoms at the boundary	437
R.3	Partitioning in LAMMPS[232]	438
T.1	Ackland Mendelev	477
T.2	Buckingham	478
T.3	Lennard Jones	479
T.4	Mishin Density	481
T.5	Morse	482
T.6	Polynomial Embedding	483
T.7	Spline	486
T.8	Slater 4s	487
T.9	Zero	489
T.10	Polynomial knot to knot spline	490
U.1	FePd Structure 01	492
U.2	FePd Structure 02	492
U.3	FePd Structure 03	493
U.4	FePd Structure 04	493
U.5	FePd Structure 05	494
V.1	Binary Alloy Potential for Fe-Pd	496
V.2	Fe-Pd - a measure of how well the potential fits to first principles calculations	497
V.3	Equation of State plots from this potential for FCC Fe	497
V.4	Equation of State plots from this potential for FCC Fe	498
V.5	Ravindran et al[130] distortion-energy plots for FCC Fe	499
V.6	MSKP elastic constant strains, where the solid line is the known value and the points the values for this potential - FCC Fe	499
V.7	Equation of State plots from this potential for FCC Pd	500
V.8	Ravindran et al[130] distortion-energy plots for FCC Pd	501
V.9	MSKP elastic constant strains, where the solid line is the known value and the points the values for this potential - FCC Pd	501
V.10	Equation of State plots from this potential for FCC Pd	502
V.11	Ravindran et al[130] distortion-energy plots for FCC Pd	503
V.12	MSKP elastic constant strains, where the solid line is the known value and the points the values for this potential - BCC Fe	503
W.1	Binary Alloy Potential for Fe-Ru	505
W.2	Fe-Ru - a measure of how well the potential fits to first principles calculations	506

W.3 Equation of State plots from this potential for FCC Fe	506
W.4 Equation of State plots from this potential for FCC Fe	507
W.5 Ravindran et al[130] distortion-energy plots for FCC Fe	508
W.6 MSKP elastic constant strains, where the solid line is the known value and the points the values for this potential - FCC Fe	508
W.7 Equation of State plots from this potential for FCC Ru	509
W.8 Ravindran et al[130] distortion-energy plots for FCC Ru	510
W.9 MSKP elastic constant strains, where the solid line is the known value and the points the values for this potential - FCC Ru	510
W.10Equation of State plots from this potential for BCC Fe	511
W.11Ravindran et al[130] distortion-energy plots for FCC Ru	512
W.12MSKP elastic constant strains, where the solid line is the known value and the points the values for this potential - BCC Ru	512

List of Tables

1.1	Bulk, shear, Young's modulus comparison: Zirlo and austenitic stainless steel	5
1.2	Simulation sizes by PKA energy approximated using SRIM[32]	12
2.1	Neutron categories by energy range [33]	16
2.2	Coarse radiation weighting factors[42]	20
2.3	Cohesive Energies [43]	21
2.4	Average recoil energy of Nickel PKAs[44]	21
3.1	Radioactive products and their half lives for PGM doped 304SS	36
4.1	University of Birmingham Cyclotron Ion Beams. The projectiles per area is based on a 0.8cmx0.8cm beam aperture.	38
4.2	Examples of neutron sources, energies and flux, with the MC-40 as a reference for protons	41
5.1	Seven Crystal Classes	54
5.2	Example bulk modulii for three metals	55
5.3	Caesium 2BEAM parameters	67
5.4	Bravais lattice vector length and angle relationships	81
5.5	Experimental vs LSDA vs GGA for $TiSi_2$ [130]	93
5.6	Experimental vs LSDA vs GGA vs GGA PBESOL for $LaAlO_3$ [174]	93
5.7	Experimental vs LSDA vs GGA - the DFT values were computed with a PWscf[177] using a 2x2x2 cell, 7x7x7 kpoints and $ecutwfc=50$	94
6.1	Half life of isotopes in the Po-216 decay chain	107
6.2	Parameters used for the decay equation vs numeric calculation comparison	107
6.3	Beam Characteristics of the Scanditronix MC-40	114
6.4	Parameters used to estimate Co55 931keV gamma activity	119
6.5	Estimation of the Cobalt-55 931keV gamma activity based on the TENDL-2009[104], TENDL-2017[196], TENDL-2019[197] data files, EXFOR experimental data [198] and TALYS generated data files[199].	119
6.6	Comparison of proton irradiated iron Co55 931keV gamma rates	123

6.7 Comparison of proton irradiated iron up to 100DPA at a range of proton energies. The first set of measurements ¹ are calculated values at the point the proton beam is turned off. The second set ² are calculated after 1 week of cooling.	123
6.8 Gamma emissions measured from a 0.5mm thick sample of Molybdenum irradiated with 13MeV protons at 5 microamps for 1,500 seconds	125
6.9 A comparison between measured and estimated peaks from a 0.5mm thick sample of Mo irradiated with 13MeV protons at 5 microamps for 1,500 seconds	127
6.10 A comparison between measured and calculated peaks from a 0.5mm thick sample of Mo irradiated with 13MeV protons at 5 microamps for 1,500 seconds	129
 7.1 Predicted lattice parameters based on the density, atomic number and type of structure	132
7.2 Predicted lattice parameters based on the density, atomic number and type of structure	135
7.3 Experimental vs LSDA vs GGA - the DFT values were computed with a PWscf[177] using a 2x2x2 cell, 7x7x7 kpoints and ecutwfc=50	138
7.4 Chromium properties with and without collinear spin	139
7.5 BCC and FCC Iron properties with and without collinear spin	140
7.6 DFT Settings - pseudopotentials, ecutwfc, ecutrho, k-points, smearing	141
7.7 DFT Settings - other parameters explicitly set in PWscf	141
7.8 Relaxed energies calculated in Quantum Espresso - U_{nn} is the entry in the 9 element matrix that represents the three basis vectors of the cell, where values not specified here are equal to zero.	143
7.9 DFT computed FCC Fe parameters for fitting the potentials in this work	146
7.10 DFT computed FCC Pd parameters for fitting the potentials in this work	147
7.11 DFT computed FCC Ru parameters for fitting the potentials in this work	147
7.12 DFT calculated energies per atom	148
7.13 Relaxed energies for 32 atoms, using 9x9x9 k-points, and 256 atoms, using the gamma point. These were used with the known cohesive energy to compute the offset energy.	149
7.14 A summary of the configurations used for fitting the interatomic potentials. The Fe-Pd and Fe-Ru fitting also included the configurations for the pure elements, although they were not required when fitting the cross potential. Random configurations include those with slightly perturbed atom positions as well as those with slightly perturbed lattice parameters.	153
7.15 Applying the Sheng Al potential, developed for FCC crystals, to the BCC structure. Elastic constants are calculated using the method by Mehl et al with values from the method by Ravindran et al in brackets.	159
7.16 Applying the Mendelev 2003 potential[143], developed for BCC crystals, to the FCC structure. Elastic constants are calculated using the method by Mehl et al with values from the method by Ravindran et al in brackets.	159
7.17 Applying the Ackland 1997 potential[222], developed for BCC crystals, to the FCC structure. Elastic constants are calculated using the method by Mehl et al with values from the method by Ravindran et al in brackets.	159
7.18 A summary of bulk properties from these potentials. The bulk modulus is computed from the Birch-Murnaghan equation of state (a) and an average of the Reuss and Voight bulk modulus (b). The FCC properties of Fe are compared to those predicted with two existing potentials designed for BCC Fe to highlight the issue of transferability and the need for new potentials.	163

A.1 Useful Laplace transforms	196
B.2 Chromium-49 - numeric vs analytic calculated radioactivity	211
B.4 Nickel-66 - numeric vs analytic calculated radioactivity	212
B.6 Caesium-125 - numeric vs analytic calculated radioactivity	213
B.8 Bismuth-213 - numeric vs analytic calculated radioactivity	214
B.10 Polonium-216 - numeric vs analytic calculated radioactivity	215
B.12 Radon-218 - numeric vs analytic calculated radioactivity	217
E.1 Simulation settings for 36 MeV proton irradiated iron	298
E.2 Simulation settings for 13 MeV proton irradiated molybdenum	315
E.3 Simulation settings for 13 MeV proton irradiated steel	324
H.1 Simple cubic configuration	362
H.2 Body centered cubic configuration	363
H.3 Face centered cubic configuration	364
H.4 Zincblende configuration	364
I.1 Independent elastic constants for a cubic crystal[223]	367
I.2 Independent elastic constants for an orthorhombic/orthotropic crystal[223]	368
I.3 Independent elastic constants for a monoclinic crystal[223]	368
I.4 Independent elastic constants for a triclinic crystal[223]	369
I.5 Calculation of elastic constants using the equation of state, a volume conserving orthorhombic strain and volume conserving monoclinic strain.[131][124]	369
I.6 Calculation of orthorhombic elastic constants by Ravindran et al.[dfttisiravindran]	370
I.7 Calculation of elastic constants for a cubic crystal[155]	371
I.8 Calculation of elastic constants for a cubic crystal[155], $u = (1 - \sigma^2)^{-\frac{1}{3}}$	372
N.1 Aluminium DFT settings	394
N.2 FCC Aluminium: Experimental Values vs DFT Calculated Variables	396
N.3 Iron DFT settings	401
N.4 BCC Iron: Experimental Values vs DFT Calculated Variables	405
N.5 Iron DFT settings	406
N.6 Iron FCC: DFT Calculated Properties	409
N.7 Palladium DFT settings	410
N.8 FCC Palladium: Experimental Values vs DFT Calculated Properties	413
N.9 Ruthenium DFT settings	414
N.10 Ruthenium FCC: DFT Calculated Properties	417

O.1 Improving convergence SCF calculations of defects in Iron	421
V.1 The lattice parameter, cohesive energy and bulk modulus were determined by fitting the Birch-Murnaghan equation of state. The bulk modulus is computed from the Birch-Murnaghan equation of state (a) and the average Reuss and Voight Bulk Moduldus (b). The elastic constants computed for the potential in stiffness (1) were computed using the method by Ravindran et al[130] and in stiffness (2) by Mehl et al[124][131].	498
V.2 The lattice parameter, cohesive energy and bulk modulus were determined by fitting the Birch-Murnaghan equation of state. The bulk modulus is computed from the Birch-Murnaghan equation of state (a) and the average Reuss and Voight Bulk Moduldus (b). The elastic constants computed for the potential in stiffness (1) were computed using the method by Ravindran et al[130] and in stiffness (2) by Mehl et al[124][131].	500
V.3 The lattice parameter, cohesive energy and bulk modulus were determined by fitting the Birch-Murnaghan equation of state. The bulk modulus is computed from the Birch-Murnaghan equation of state (a) and the average Reuss and Voight Bulk Moduldus (b). The elastic constants computed for the potential in stiffness (1) were computed using the method by Ravindran et al[130] and in stiffness (2) by Mehl et al[124][131].	502
W.1 The lattice parameter, cohesive energy and bulk modulus were determined by fitting the Birch-Murnaghan equation of state. The bulk modulus is computed from the Birch-Murnaghan equation of state (a) and the average Reuss and Voight Bulk Moduldus (b). The elastic constants computed for the potential in stiffness (1) were computed using the method by Ravindran et al[130] and in stiffness (2) by Mehl et al[124][131].	507
W.2 The lattice parameter, cohesive energy and bulk modulus were determined by fitting the Birch-Murnaghan equation of state. The bulk modulus is computed from the Birch-Murnaghan equation of state (a) and the average Reuss and Voight Bulk Moduldus (b). The elastic constants computed for the potential in stiffness (1) were computed using the method by Ravindran et al[130] and in stiffness (2) by Mehl et al[124][131].	509
W.3 The lattice parameter, cohesive energy and bulk modulus were determined by fitting the Birch-Murnaghan equation of state. The bulk modulus is computed from the Birch-Murnaghan equation of state (a) and the average Reuss and Voight Bulk Moduldus (b). The elastic constants computed for the potential in stiffness (1) were computed using the method by Ravindran et al[130] and in stiffness (2) by Mehl et al[124][131].	511

Chapter 1

Introduction

Several Gen III+ nuclear reactors have been proposed for construction in the UK and Gen IV nuclear reactors are being researched and developed. At the time of writing Hinkley Point C is under construction and Sizewell C has had government approval. However, much more must be done to ensure the energy security of our nation. Well established reactor types and exciting new designs could contribute to solving this problem, but challenges to materials science must be overcome.

New materials are required to withstand the extreme conditions in and around the core of these reactors. Austenitic stainless steels have been an important structural material in the industry, and may continue to be so, providing the problem of inter granular stress corrosion cracking (IGSCC) can be addressed. Doping these steels with platinum group metal (PGM)s has been seen to reduce IGSCC, but the effects on corrosion resistance are unknown for these steels when irradiated by a radiation field.

1.1 Motivation

Mass produced steels are not perfect repeating crystals, but are made up of small grains. Cr is added to steel to make stainless steel, and this is more resistant to corroding than steel. When steel is in a nuclear reactor, it will have to perform under extreme conditions, such as:

- high temperatures
- strain cause by high pressures and radiation induced swelling
- radiation damage and strains resulting from this damage
- a changing environment (radiolysis, isotope transmutation)
- corrosive environments while in the reactor
- corrosive environments out of the reactor (e.g. fuel cladding in storage)

Radiation damages the steel in a number of ways including directly knocking atoms out of place within the lattice, changing the environment (e.g. radiolysis) and transmuting isotopes. An example of the latter is a neutron transmuting an Fe atom into a Co atom.

In time, the radiation damage causes the percentage of Cr at the grain boundaries to drop, and as it falls the steel loses its protection from corrosion at the boundary between the grains it is made of.

This work is divided into two parts.

1.1.1 Part 1: Activity Computer Program

The materials must be tested before being used in a nuclear reactor. One way to do this would be to place samples of the steel into a test reactor. This is expensive and, as a by-product, the steel sample becomes radioactive. It is difficult to create a large number of neutrons, but it is much easier, and cheaper, to create a beam of protons. Protons can be accelerated in a machine such as the Cyclotron at the University of Birmingham.

The damage that protons cause to a sample is not precisely the same as that caused by neutrons, but it is a cheaper alternative and is a trade-off between the cost and results. One side effect that proton irradiation shares with neutron irradiation is the creation of radioactive waste.

The first part of this work investigates exactly how radioactive a sample becomes when irradiated with a proton beam. An existing equation, named after Mathematician Harry Bateman[1], was modified and a computer program was created to perform the calculation. The user inputs the constituent elements that make up the material, the ratio of these elements, and the irradiation settings. The program then estimates how radioactive the sample will be and the predicted gamma energies.

1.1.2 Part 2: Iron-Palladium and Iron-Ruthenium Potential

The addition of Cr to make stainless steel is not the only way to make a steel resistant to corrosion. Adding metals such as Mo, Pd and Ru to steel can increase the resistance to corrosion, but Mo is several hundred times the cost of Fe ore, and Pd is thousands of times as expensive.

Simulating radiation damage using a computer is now a feasible and sensible way to investigate how these materials will be affected by radiation damage, and the simulations may reveal insights that experiments are not able to show, either because they happen on too small a timescale or within the material at too small a length scale.

Key to the simulations success is being able to accurately calculate how the atoms interact with one another. The second part of this work concentrates on deriving a mathematical description of how Pd & Fe and Ru & Fe atoms interact with one another, which would then allow future simulations of steel with small amounts of Pd or Ru added to it.

Radiation causes Cr to be depleted at the grain boundary. If these simulations go on to show that Pd and Ru are not depleted, it would suggest that the corrosion resistance is maintained despite the decrease in the amount of Cr at the grain boundary due to radiation damage.

1.2 An Approaching Energy Gap for the UK

Since Calder Hall, the first commercial nuclear power plant, opened in 1956, the demand on electrical power generation in the UK has tripled (fig 1.1). There is now a reliance on cheap and clean power from nuclear reactors as these have provided a quarter of our electricity until recently.

There is an obvious concern that within the next ten years the UK will lose a sizeable proportion of its electricity generation capabilities. There are increasing pressures on countries to reduce their carbon dioxide output, so the gap created by ageing nuclear plants and coal power needs to be filled.

1.2.1 Why choose Nuclear Power?

As a civilization we need energy, whether it's in the form of electricity or stored chemically, and whilst we are becoming more efficient at using that energy, the demand for it will remain (unless something drastically

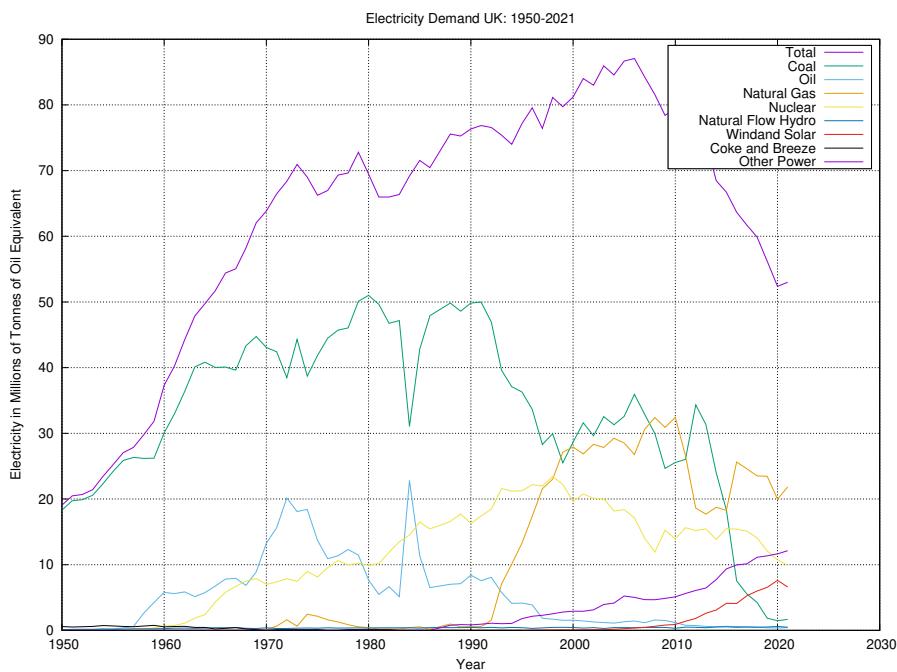


Figure 1.1: Electricity in Millions of Tonnes of Oil Equivalent[2]

changes our society). There is a choice between burning fossil fuels or bio fuels, using energy from the Sun, wind, ground, oceans or rivers and finally using the energy of the nucleus, whether by fission or fusion.

Each has its drawbacks and advantages. Renewable energies are unreliable; wind power only provides energy when there is a wind, and if the wind is too strong, they must be shut down or risk damage. The Sun is only available for a portion of the day, and the duration and intensity change with the seasons, not to mention the impact of clouds on solar power. Renewable sources are not very good at responding to demand; there isn't a button to turn up the power of the Sun, or increase the velocity of the wind when the national grid demands it. If we were completely reliant on renewable energies, there would either need to be an efficient way to store energy on a large scale, or many more solar, wind, tidal and hydroelectric power stations than would be needed to produce excess energy. Whilst the energy source may be free, turbines, solar panels, dams and so on require maintenance, so an excess of these would be wasteful and costly.

Fossil fuels release carbon dioxide, sulphur dioxide, naturally occurring trace radioactive elements and other pollutants into the atmosphere. These power plants are useful because the fuel is cheap, but the waste is put back into the environment without much processing and the energy produced may be varied to meet the demand of the national grid.

Nuclear power has a bad reputation with the public, in part caused by accidents such as the Windscale fire, Three Mile Island, Chernobyl, and Fukushima. There have been examples of poor handling of nuclear waste in the past, such as the legacy storage ponds at Sellafield, and long term storage in geologically stable areas is something that hasn't been achieved to store the existing waste, let alone waste created in the future.

Modern nuclear power plants are very expensive to build, with the proposed 3.2GWe Sizewell C power plant expected to cost £20 billion or more [3]. When compared to £0.5 billion for the 0.884GWe Carrington CCGT Power Station [4], the initial costs are £0.57 billion per GWe for CCGT compared to £6.25 billion per GWe for Nuclear.

There is an effort around the world of countries aiming to reduce the production of carbon dioxide as a result of burning fossil fuels. When a power plant is operational, the carbon dioxide produced by nuclear power is negligible when compared with gas, oil and coal power plants.

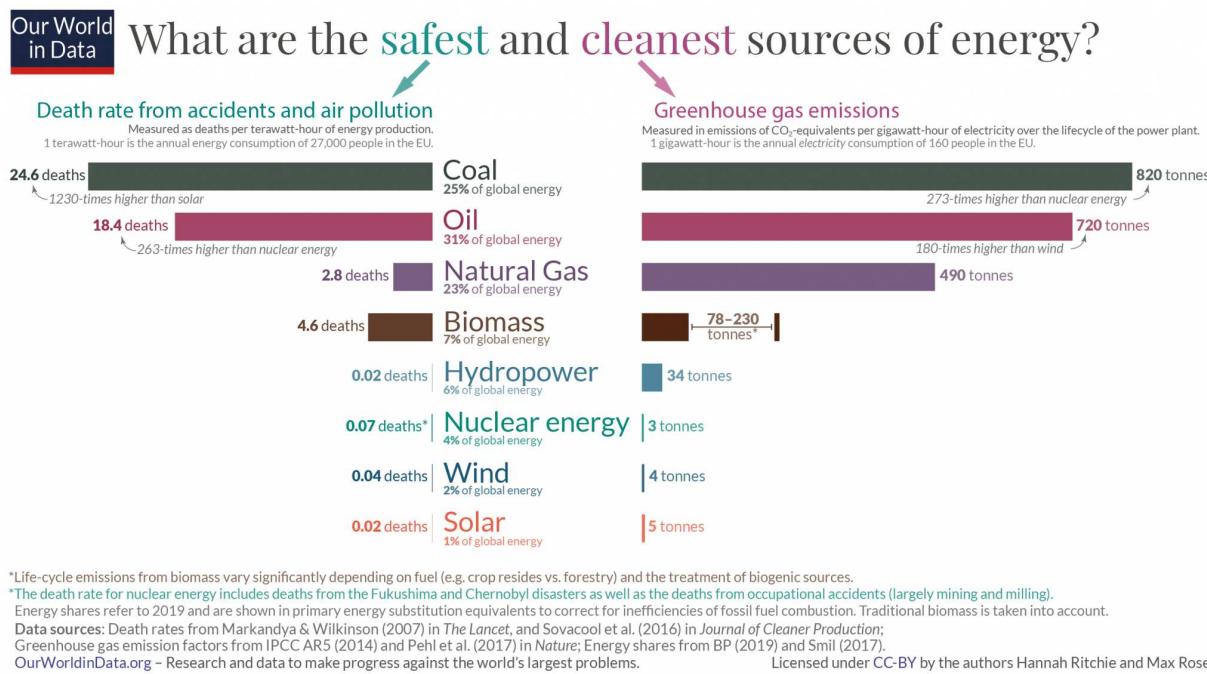


Figure 1.2: Safest and cleanest sources of energy [5] - deaths per terawatt hour and tonnes of CO₂ per gigawatt hour

Despite its reputation amongst the public, nuclear is much safer than fossil fuels and is on par with renewable energies. The pollution caused by fossil fuels affects us all and is responsible for many deaths each year, causing more misery than nuclear power ever has (fig. 1.2). Coal and oil are responsible for 350 and 260 times as many deaths as nuclear, respectively.

As a species, we have used combustion of chemicals to generate heat for thousands of years, culminating in high efficiency CCGT power stations, with efficiencies above 60%. We are at the beginning of our journey with nuclear power, with many new and innovative designs proposed, where safety to plant workers and the general public are a priority.

Future designs may help to tackle the problematic waste generated in the past, and by using fertile material to produce fissionable elements other than $^{235}_{92}U$, the amount of fuel available to reactors will increase.

1.2.2 Proposed Generation III+ Nuclear Power Plant Designs

Areva EPR

The 1.6GW Areva EPR design has four primary loops transferring heat by pressurised water from the reactor to heat exchangers. It requires U enriched to 5% $^{235}_{92}U$ in the form of U Oxide Pellets. The inlet temperature is 568.75K and the outlet temperature is 602.95K.

There are 241 fuel assemblies, each containing 265 fuel rods giving a total of 63865 rods. The fuel rod cladding is made from 316 stainless steel and has an inner diameter of 7.72mm and an outer diameter of 9.68mm. The materials used to construct the control rod drive mechanisms includes 410 stainless steel and 304 stainless steel.

There are two EPR reactors planned for the proposed Sizewell C power plant and they will be designed to produce a combined power output of 3.2 GWe. The government has authorised planning consent for the power station and hopefully construction will start within the next couple of years.

Westinghouse AP1000

The 1.1GW Westinghouse AP1000 design is a pressurised water reactor with two primary loops transferring heat from the reactors to heat exchangers. An enriched UO_2 fuel, up to 5% ^{235}U , is clad in Zirlo (a proprietary Zr alloy). Zirlo is an alternative cladding material to 316 stainless steel, and as a cladding material is absorbs fewer neutrons than steel cladding. The inlet temperature is $552.55K$ and the outlet temperature is $597.85K$ which is close to the temperature range of the EPR. The composition of the control rod absorber material includes 304 stainless steel.

Property	316SS	Fe 20Cr 8Ni 1Mo	Zirlo (Hill Approximation)
Bulk Modulus (GPa)	164.9 [6]		98.5 [7]
Shear Modulus (GPa)	74.6 [6]		33.2 [7]
Young's Modulus (GPa)	194.3 [6]		89.7 [7]
Poisson's Ratio	0.30 [6]		0.35 [7]

Table 1.1: Bulk, shear, Young's modulus comparison: Zirlo and austenitic stainless steel

1.2.3 Generation IV

Goals of Generation IV Reactors

The GenIV International Forum has put forward four main goals for this next generation of nuclear power[8]:

- sustainability
- safety and reliability
- economics
- proliferation resistance and physical protection

The selection of known materials, and the development of new materials, will play a key part in all four of these goals.

Carnot's Theorem

Whatever the motivation, whether it is to increase profits or to supply energy at greater amounts and for less cost to the consumer, increasing the efficiency of energy production is critical. Solar panels are continually being improved to edge closer to their theoretical limit and wind turbines are being constructed larger and with more advanced materials to extract as much energy from the wind as possible.

In the same way, power plants that use heat are constantly being developed to improve efficiency. In the nineteenth century Carnot showed that the maximum possible efficiency of a heat engine is determined by the difference in temperature between the heat reservoirs.[9]

$$\eta_{max} = 1 - \frac{T_c}{T_h} \quad (1.1)$$

To increase maximum efficiency, the temperature difference should be increased, and this leads to higher temperature reactors. There will be a trade off between the increased temperature, the ability of the materials

to withstand the temperature, health and safety considerations, the lifetime of components, the effect on the coolant and more.

The first generation of reactors in the UK were the gas cooled Magnox reactors. With core temperatures of approximately $620K$ [10], the thermodynamic efficiency was relatively poor. This was limited by the MgO cladding, which was in turn selected due to the fuel. Combined cycle gas turbines have much higher temperatures within their turbines, and the temperature of the steam within the secondary steam turbine can reach $850K$ [11], significantly higher than in a Magnox reactor.

The second generation of power plants, particularly in the UK, included the advanced gas reactor that used CO_2 as a coolant. This reactor design increased temperatures to over $870K$, and thus the maximum possible thermodynamic efficiency was increased.

Several Gen IV reactors have designed operating temperatures that exceed those of the AGR, and this brings a new set of challenges to overcome.

Fast Reactors

Examples of thermal neutron reactors include Magnox, AGR, LWR and Candu reactors. These reactors contain moderators, designed to slow neutrons, decreasing their energy to thermal temperatures and leveraging the large neutron fission cross section of ^{235}U . The cross section for fast neutrons and ^{235}U is in the region of 1 barn, but thermal neutrons and ^{235}U have a fission cross section of over 1,000 barns.

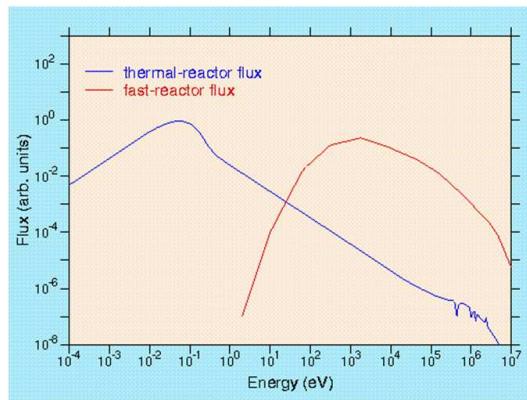


Figure 1.3: Typical reactor flux - thermal and fast spectra [12]

Natural U contains 0.7% ^{235}U and 99.3% ^{238}U [13]. The thermal fission cross section for ^{238}U is small for thermal neutrons, and can be measured in the microbarn to millibarn range, but it has a much higher fission cross section with fast neutrons with a cross section closer to 1 barn for neutrons with an energy of 1MeV or more. The neutron flux in a fast reactor is spread over a higher energy band than that of a thermal neutron reactor (fig. 1.3).

Fast neutron reactors have been tested and used since the 1950s, and there are several Gen IV reactor designs based on this approach. Given the much larger percentage of ^{238}U in natural U, Fast Neutron Reactors have a larger stockpile of fuel. They also breed fissile isotopes, ^{239}Pu and ^{241}Pu . This has its advantages and disadvantages. The advantage is clear, as it produces fissile fuel as it runs, but the creation of fissile materials is a concern as these isotopes could be used in nuclear weapons.

Lead-Cooled Fast Reactor

Light elements such as the hydrogen in water molecules, or carbon in a graphite stack, allow neutrons to efficiently lose energy. When neutrons scatter inelastically with heavier elements, such as lead, they lose a much

smaller percentage of their energy (fig. 1.4).

The Lead-Cooled Fast Reactor (LFR) uses lead as the coolant. Lead has a low reaction cross section and, because of its mass relative to a neutron, the kinetic energy lost in collisions is low, keeping more neutrons in the fast energy group. As the coolant is a liquid, and the boiling point of lead is over $1,970K$, the system will run at atmospheric pressures and any problems associated with void formation due to boiling are removed. The molten lead will act as a gamma shield and it will be chemically less reactive than the coolant in a Sodium-Cooled Fast Reactor (SFR) or Supercritical-Water-Cooled Reactor (SCWR).

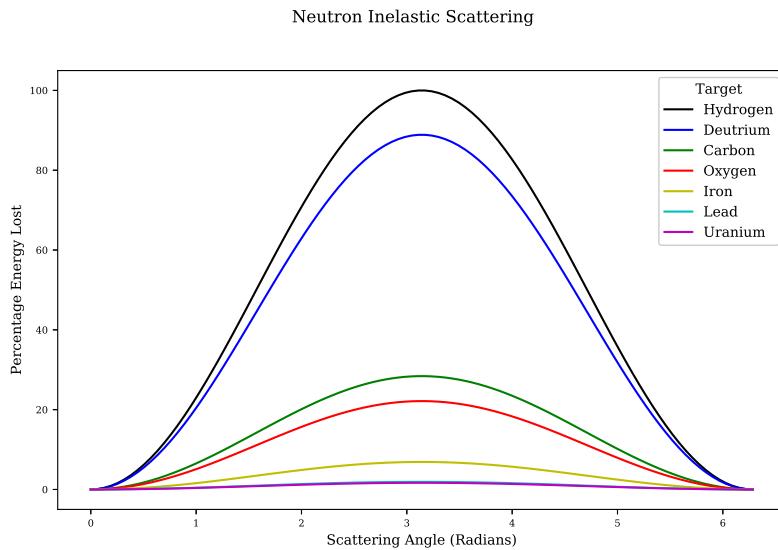


Figure 1.4: Inelastic scattering - scattering angle vs energy loss for a range of target atoms - generated with a python script using the equations contained in Duderstadt and Hamilton: Nuclear Reactor Analysis[14]

There are several disadvantages to using lead as a coolant. The melting point of lead is $600K$, so the reactor would need to be heated first for the coolant to become a liquid. This also restricts the lowest temperature and thus the maximum efficiency possible that may be extracted from the system without the coolant forming a solid. The density of lead also poses a problem for the structure needed to support the reactor.

European Lead Fast Reactor (ELSY) is a 600MWe Lead-Bismuth eutectic cooled fast neutron reactor. It has a lower melting point than lead, but there are concerns for the transmutation of Bismuth to Polonium. The 9,000 tonne molten lead coolant and the high flux of fast neutrons[15] will be challenging conditions to overcome.

Very-High-Temperature Reactor

Traditional nuclear power plants, as well as coal and the secondary cycle of CCGT plants, boil water to drive turbines. Very-High-Temperature Reactor (VHTR) designs use thermal neutrons and a fissile fuels. Designs plan to have outlet temperatures of up to $1,270K$ [16]. With a helium coolant, there are several options.

- directly drive turbines
- heat water to create steam and drive turbines
- use the high temperature to help create hydrogen, and combine with steam driven turbine

Hydrogen may be extracted from high temperature water either by a thermo-chemical, high temperature electrolysis or a hybrid process[17]. Each process benefits from the very high temperatures provided by the predicted outlet temperatures (fig 1.5).

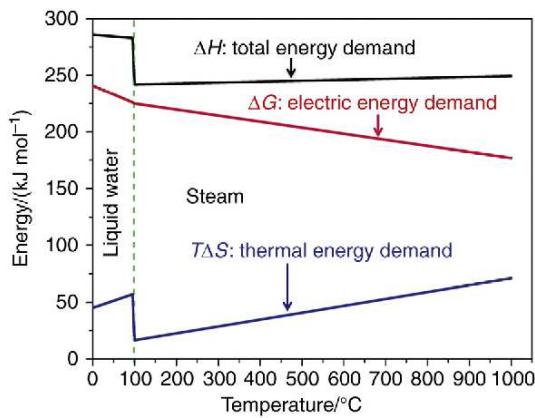


Figure 1.5: Energy demand to produce hydrogen by electrolysis as temperature changes[18]

The Pebble-Bed Reactor (PBR) is a type of VHTR where spherical fuel pellets are held in a hopper shaped reactor. Temperatures within the core may reach 1,770K, and this will pose a challenge for materials scientists.

Gas-Cooled Fast Reactor

The Gas-Cooled Fast Reactor (GFR) design is similar to the VHTR, but rather than use thermal neutrons, it will use fast neutrons. The design will use a ceramic core and have outlet temperatures up to 1120K. This will give similar options to the VHTR where high temperatures would be used to assist in the production of hydrogen, or it would be used to power a turbine with a secondary steam powered circuit similar to that of a CCGT.

There are challenges ahead, including the development and testing of advanced materials that can withstand these temperatures while under high energy neutron flux. However, unlike LFRs, SFRs and SCWRs, the coolant is chemically inert and isn't transmuted by neutrons removing the risk of the coolant becoming radioactive.

Sodium-Cooled Fast Reactor

Memories of small amounts of sodium, kept under oil in school chemistry labs, being carefully cut and prepared ahead of a violent reaction with water, probably come to mind at the first mention of an Sodium-Cooled Fast Reactor (SFR). It may not be the first choice of an element to use as a coolant, but it does have advantages. In fact Admiral Rickover once remarked “if the oceans were filled with liquid sodium, then some crazy scientist would want to build a water-cooled reactor” when the idea of a sodium cooled reactor in a submarine was proposed[19].

Sodium has a low melting point, of just under 370K, and a boiling point of 1,150K. With an operating temperature of up to 820K, the reactor would be able to run at atmospheric pressure and without the issues associated with the coolant boiling. The density of Na is 0.971gcm^{-3} and comparing this to the LFR the coolant would be 12 times less dense and this would have advantages when designing the structure.

There have been several SFRs built and operated using this technology, such as the Russian BN-600 reactor. This particular reactor has a sodium primary loop which heats a secondary sodium loop which in turn heats a third loop for water and steam.

Supercritical-Water-Cooled Reactor

At 647K and a pressure of 22.1MPa (approximately 218 atmospheres), water becomes supercritical (fig. 1.6) and at this point it is neither a liquid or a gas[20]. The current generation PWR and BWR operate at approximately

$501 - 598K$ at 15.2MPa [21] and $551 - 560K$ at 7.1MPa [22] respectively.

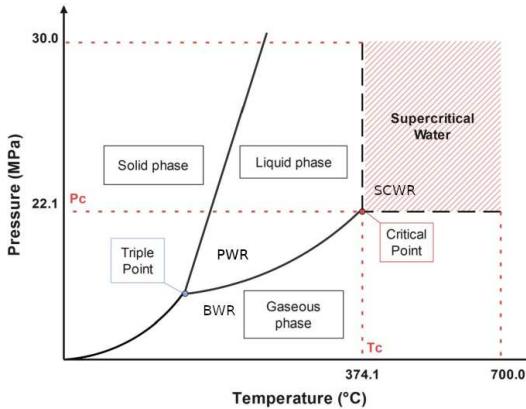


Figure 1.6: Phases of water: pressure vs temperature [23]

Supercritical water exists above $647K$ and 22.1MPa , and in this state water has a higher thermodynamic efficiency. The design of the nuclear power plant is also simplified as there is no phase change of the water, so a condenser is not needed. The Supercritical-Water-Cooled Reactor (SCWR) is the only Gen IV reactor design that uses water as the coolant[24]. The economic benefits have already been seen in SCW fossil fuel power stations, and it is incorporated in Gen IV water cooled fast and thermal reactors.

There are side effects to using supercritical water. It is an oxidant and is now being considered to “burn” organic waste in future waste processing plants. The combination of supercritical water chemistry and irradiation damage must be considered, as well as higher temperature and pressure, when considering the materials used by a SCWR.

Molten Salt Reactor

MSRs have been operated for over 50 years. Chemically, salts are very stable, and this has obvious safety benefits over reactors like the SFR. The coolants are fluoride salts that have high boiling points which gives the added safety protection of being able to operate at atmospheric pressure, unlike a PWR where a pressure vessel is needed.

The fuel is dissolved into the molten salt. There are no solid fuel rods to place into the core, remove or reprocess. The molten salt is processed on the site to remove poisons, waste and add new fuel.

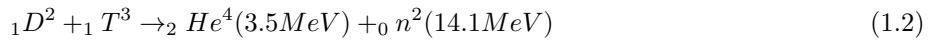
MSRs may use either thermal or fast neutrons and a range of fuels including Th. This is particularly interesting as Th is three times as abundant as U and would increase the fuel available to us for the clean production of electricity by nuclear power.

A safety feature that takes advantage of the fuel being dissolved in the molten salt are large drain tanks under the reactor. A plug is designed to melt if it reaches a certain temperature and this would quickly drain the molten salt from the reactor into large and cold drainage tanks. The reactivity would drop and the fuel would cool reverting it to a solid salt.

Salts considered in various designs are chlorides, nitrates and fluorides. Corrosion of the reactor due to the molten salts is a concern. Elements that provide protection to corrosion, such as Cr, are prone to dissolution into the molten salt[25]. Where the fuel is suspended in the molten salt as Tristructural Isotropic (TRISO) fuel particles, formation of chromium-carbide precipitates may be an issue.

Experimental Fusion Reactors

Nuclear Fusion is a very attractive technology and could be the answer to all of our energy problems. Much work is being invested in developing this technology and the International Thermonuclear Experimental Reactor (ITER) has been designed to output more energy than is required to start the fusion reaction. The process of fusion combines two isotopes of hydrogen and creating helium, fast neutrons (eq. 1.2) and an excess of energy. As neutrons have no charge, they can penetrate shielding causing damage as they lose energy through nuclear interactions. Any atoms they interact with have a chance to capture the neutron and become unstable.



The fast neutron spectrum for fission reactors ranges from a few eV to a few MeV, whereas the neutrons in a fusion reaction have 2-3 times more energy than the most energetic neutrons from fast fission. Engineers must develop materials to construct components that will be resilient to this damage, while having a low reaction cross section and being able to withstand other extreme conditions within the reactor.

Unfortunately, nuclear fusion as a commercial method of creating energy, is often said to be 30 years away, and this has been the case for some time now. It is a lot to ask, recreating a process that occurs within a star, in a safe and steady manner. Once this problem is solved, it could mark a turning point for human civilization.

1.3 Damage to Nuclear Core Components by Neutrons

Radiation damage in a reactor is caused by several projectiles. Fission fragments are heavy, contain charged particles and lose energy close to their source. Gamma rays excite electrons, promoting them to higher energy levels, or ionise the atoms where there is enough energy to eject electrons completely from atoms. They do not have the momentum to knock atoms out of place. Fast neutrons, however, may impart a great deal of momentum to a target atom and, as they are neutral, travel much further into a material than fission fragments. With enough energy, neutrons cause large damage cascades within the material.

One fuel source for many of the Gen III+ and Gen IV reactors is ^{235}U , whether as enriched U or otherwise. The neutron(s) released by the fission of ^{235}U atoms have a spectra of energy (fig. 1.7) which may roughly be split into four categories: cold (below 0.025eV), thermal (0.025eV), slow and intermediate (above 0.025eV and below 1MeV) and fast (1MeV and above).

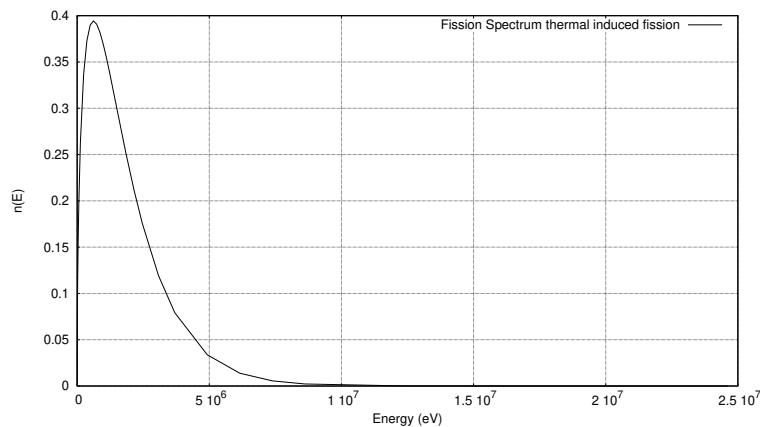


Figure 1.7: Neutron energy spectrum from fission generated from the JEFF 3.1.1 data file[26]

Thermal and slow/intermediate neutrons cause damage in their own particular way, as they are captured by and transmute the atoms within the target material. Fast neutrons have enough energy to create a great deal

of damage, whereby they transfer kinetic energy to an atom in the material that becomes the primary knock-on atom (PKA) in a cascade of atoms through the material. Neutrons can travel deep into a material, creating many damage cascades throughout until they lose enough energy or pass through completely.

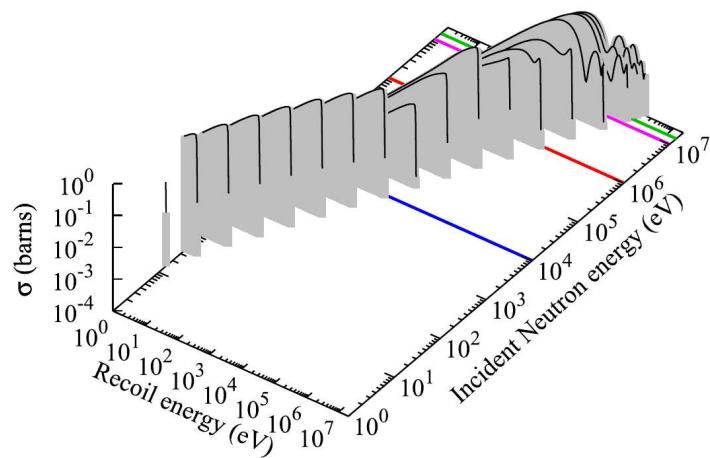


Figure 1.8: PKA energy as a function of neutron energy in Fe[27]

The intensity of neutron energies from the fission of $^{235}_{92}U$ peaks around 1MeV (fig. 1.7). The energy transferred to PKAs will vary depending on the mass of the target nucleus and scattering angle but, in Fe at least, 1MeV neutrons will create the majority of its Fe PKAs with energies ranging from 1keV to 100keV (fig. 1.8).

1.4 Radiation Damage: Replacing Neutrons with Protons

It is difficult to generate large fluxes of neutrons. Neutrons have no net charge, so they cannot be controlled in a similar way. Nuclear reactors are the most common way to create large fluxes of neutrons.

This is an expensive method, and one that may be inconvenient as the sample being tested would need to be placed inside a reactor. Rather than do this, ions may be used to replicate the damage caused by neutrons, but in a more controlled and accessible way.

Experimental reactors are not able to damage a material significantly faster than operational reactors, so a major reason for approximating with ion irradiation is the higher damage rate, with the hope of discovering potential problems before they manifest in current reactor designs[28].

The damage caused by neutrons, light ions and heavy ions differ due to their interactions with matter. Ions lose energy gradually but within a very short distance, whereas neutrons lose larger amount of energy less frequently, but over much greater distances. The PKA energies as a result are higher for neutrons than ions, however comparisons of low damage dose neutron and light ion irradiation show cluster size distribution to be independent of PKA energies[28][29].

A major side effect from the process of nuclear fission is the creation of both radioactive fission fragments and radioactive isotopes within the components and structural material of the reactor. Low energy protons are not captured as low energy neutrons would be, due to the opposing electromagnetic force between the proton and nucleus of the target atom. Once the proton energies exceed a few MeV, they have sufficient energy to transmute target nuclei.

The benefits of ion irradiation as discussed, and positive studies into the independence of damage due to PKAs in the energy range 6.6keV to 176keV[29], support its use in emulating neutron damage.

Induced radioactivity is still a problem. By virtue of detailed reaction cross section data files and decay data, the amount of radioactivity induced by both protons and neutrons may be calculated. This will assist in the selection of proton beam parameters to replicate irradiation damage, to the required DPA, whilst keeping the radioactivity at a safe level.

1.5 Simulated Material Damage

Irradiating and testing materials with both neutrons and protons has a number of drawbacks, including expensive facilities and the creation of radioactive waste. Simulations may be used to help understand the processes on a mesoscopic scale and make predictions on how the materials in question may behave inside a reactor under certain conditions.

Part of this work focuses on the damage of stainless steel by neutrons. The damage process occurs on time scales and sizes too small to capture experimentally and play back to study. The theories may not be sufficiently well understood, or are often unsolvable with current techniques. Modelling helps bridge the gap between theory and experiment.

1.5.1 Molecular Dynamics

Molecular dynamics models have been used to simulate radiation damage. Damage cascades caused by primary knock-on atom (PKA)s with energies of 10, 20 and 50KeV have been modelled in BCC W[30], whilst cascades caused by PKAs of 5, 10 and 20KeV have been simulated in BCC Fe[31].

Neutrons have a long mean path before interacting with a target when compared to ions. It would be impossible with current and foreseeable technology to simulate a neutron travelling through a material because the volume of the material, and the number of atoms and interactions between atoms, would be so large. Rather than attempt this, it is more productive to focus on individual damage cascades caused by PKAs in order to conserve computing resources.

Molecular Dynamics simulations require interatomic potentials to govern how the atoms in the simulation interact with one another. Typical simulation sizes range from thousands of atoms to millions of atoms.

Fe has a density of approximately 8.5×10^{-2} atoms per cubic angstrom, and this leads to simulation sizes between almost one million atoms and fifty million atoms for cascades with PKAs ranging from 5KeV to 50KeV (fig 1.9, table 1.2).

PKA Energy (KeV)	Volume (Ang^3)	Atoms
5	1.0×10^6	9.0×10^5
10	6.0×10^6	5.4×10^6
20	8.0×10^6	6.9×10^6
50	6.4×10^7	5.5×10^7

Table 1.2: Simulation sizes by PKA energy approximated using SRIM[32]

The interatomic potentials used to simulate a damage cascade must be able to model the very close separations during the collisions event. For very close separations, an exponential potential such as the Ziegler-Biersack-Littmark (ZBL) potential may be used, and for other separations a standard potential such as the embedded-atom method (EAM) may be used, in particular for metals.

There have been attempts to create potentials based on Physics, but the nuances are still poorly understood and there aren't any models that can be applied accurately to any material type under any set of circumstances.

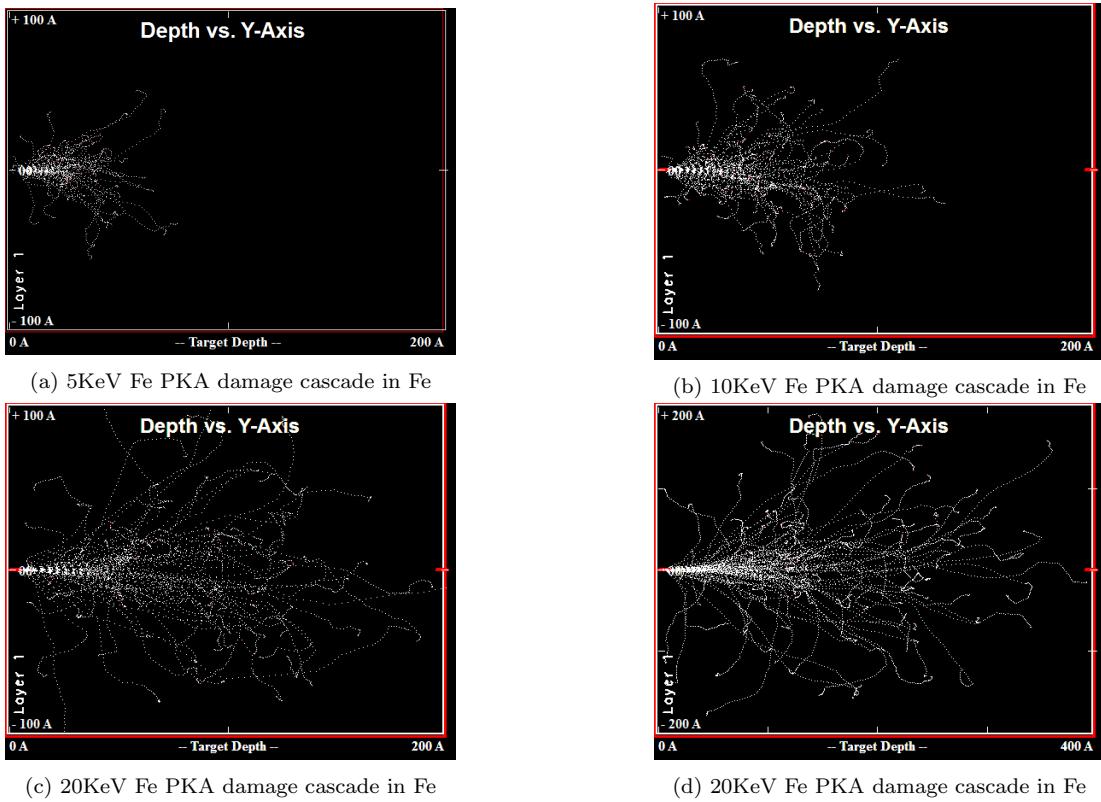


Figure 1.9: Damage cascades in Fe for a range of energies generated in SRIM[32]

A number of recent interatomic potentials, including a selection of transition metals, have been fit to match the experimental data rather than the underlying physics.

1.5.2 Density Functional Theory

In Quantum Mechanics, the Schrödinger equation and the wavefunction of a system describes all that can be known about that quantum system. Unfortunately, it is very difficult to compute when there are just several electrons in the system. Many atoms in a system makes this an impossible task with current technology.

Density Functional Theory (DFT) replaces the many electron wavefunction with many single electron wavefunctions. Rather than use the positions of all the electrons, which would result in 3^n parameters, the charge density is used. It is important to note that DFT is an exact method for ground state calculations, although in practice approximations do need to be made in order to solve problems. This is discussed in detail in section 5.7.

1.5.3 Interatomic Potentials for Molecular Dynamics

Interatomic potentials help to describe the interaction between atoms. Originally they were used for pairs of atoms and a collection of atoms on a pair by pair basis, but in the 1980s the complexity of potentials increased as they started to include the effects of the background electron density that the atoms are embedded in. This lead to the Finnis-Sinclair (FS) and embedded-atom method (EAM) type potentials.

Whilst DFT is a purer form of computational materials science, more closely linked to physics, it is computationally intensive and suitable for small systems of hundreds to thousands of atoms. By using interatomic potentials to recreate the forces and energies that would be computed by DFT, the systems may be increased in size to thousands to millions of atoms, and these calculations may be run over a time frame with hundreds or thousands of time steps.

There are a wide range of potentials, and these are typically selected based on the type of material and the required accuracy. Simple pair potentials may still be used, but angularly dependent potentials have been developed for materials with covalent bonds, and metals use FS and EAM potentials. The potentials are made up of a set of functions, and these functions change from derivation to derivation.

Chapter 2

Consequences of Ionizing Radiation

This work is concerned with investigating the damage caused to components by ionizing radiation, whilst minimizing the risk to people from the same radiation. In both cases the radiation can damage the component or cell directly or by changing the environment, for example the radiolysis of water. The interactions of ionizing radiation and the units of measure are discussed here.

2.1 Radiation Types Relevant to This Work

2.1.1 Introduction

There are three types of radiation that are useful to discuss in this work: neutrons, light ions and gamma rays. Two of these in particular, neutrons and protons, are of particular interest here due to the damage they may cause. Each of these are capable of ionizing the atoms they interact with, given sufficient energy. Higher energy and momentum radiation it is able to directly damage materials or the DNA structure of organic life that it interacts with. Lower momentum ionizing radiation may change the water chemistry of the environment or within the cells of organic lifeforms that is detrimental to life.

2.1.2 Protons and other Ions

Charged particles interact with matter through the Coulomb interaction. As a charged particle passes through matter, it may interact with both the nucleus and electrons of an atom. A sufficiently energetic ion may lose kinetic energy to electrons by either raising the electrons to higher energy levels in their atoms, or by removing electrons from atoms altogether, causing the ionization of those atoms.

Ions may also lose kinetic energy to the nucleus of an atom through elastic scattering and, where the atom is in a crystal structure, through knocking atoms in the material out of their lattice positions. Inelastic scattering is also a possibility, leaving the target nucleus in an excited state.

Knock on atoms and electrons with enough kinetic energy that have been removed from atoms (delta rays), continue the irradiation of the material whilst they have the kinetic energy available to do so.

A large proportion of the energy of a charged projectile is lost to the electrons of the target material. There is a chance, depending on the energy and type of charged projectile (and the cross section of the target nucleus), that the charged particle will overcome the coulomb potential and be captured by the nucleus.

This may result in a stable nucleus or an unstable nucleus. If it is unstable, there is a probability that it will decay releasing energy in the form of photons, nucleons or electrons. The initial ion irradiation creates sources of further radiation within the target material.

2.1.3 Neutrons

Neutrons interact with atoms in matter differently to that of protons, heavier ions and other atoms within the material, as the neutron has no overall charge. Neutrons interact with the nucleus of atoms, and this is dependant upon the target atom and the energy of the projectile neutron. The energy of neutrons may be grouped as shown in table 2.1, and as mentioned earlier, thermal neutrons are of interest to current reactor designs, and fast neutrons are of interest to several future designs.

Name	Energy Range	Velocity (ms^{-1})	Wavelength (angstrom)
Cold	0-0.025 eV	$0.0 - 2.2 \times 10^3$	> 1.8
Thermal	0.025 eV	2.2×10^3	1.8
Epithermal	0.025-0.4 eV	$2.2 \times 10^3 - 8.8 \times 10^3$	0.5-1.8
Cadmium	0.4-0.6 eV	$8.8 \times 10^3 - 1.1 \times 10^4$	0.4-0.5
Epicadmium	0.6-1.0 eV	$1.1 \times 10^4 - 1.4 \times 10^4$	0.3-0.4
Slow	1-10 eV	$1.4 \times 10^4 - 4.4 \times 10^4$	0.09-0.3
Resonance	10-300 eV	$4.4 \times 10^4 - 2.4 \times 10^5$	0.02-0.09
Intermediate	300 eV - 1 MeV	2.4×10^5	$2.9 \times 10^{-4} - 0.02$
Fast	1-20 MeV	$1.4 \times 10^7 - 6.1 \times 10^7$	$6.5 \times 10^{-5} - 2.9 \times 10^{-4}$
Relativistic	$> 20 \text{ MeV}$	$> 6.1 \times 10^7$	$< 6.5 \times 10^{-5}$

Table 2.1: Neutron categories by energy range [33]

The nuclear force holds nuclei together by exchanging pions and the types of interactions are shown in figure 2.1. It is 137 times stronger than the electromagnetic force, but because of the mass of the pion (approximately 130-140MeV) it only acts over a short range of approximately 1fm. The nuclear force is attractive but has a short-range repulsive core. It is this force that allows either a neutron or proton to be captured by a nucleus, although this system is much more complex. This leads to activation of the material and the creation of radioactive isotopes.

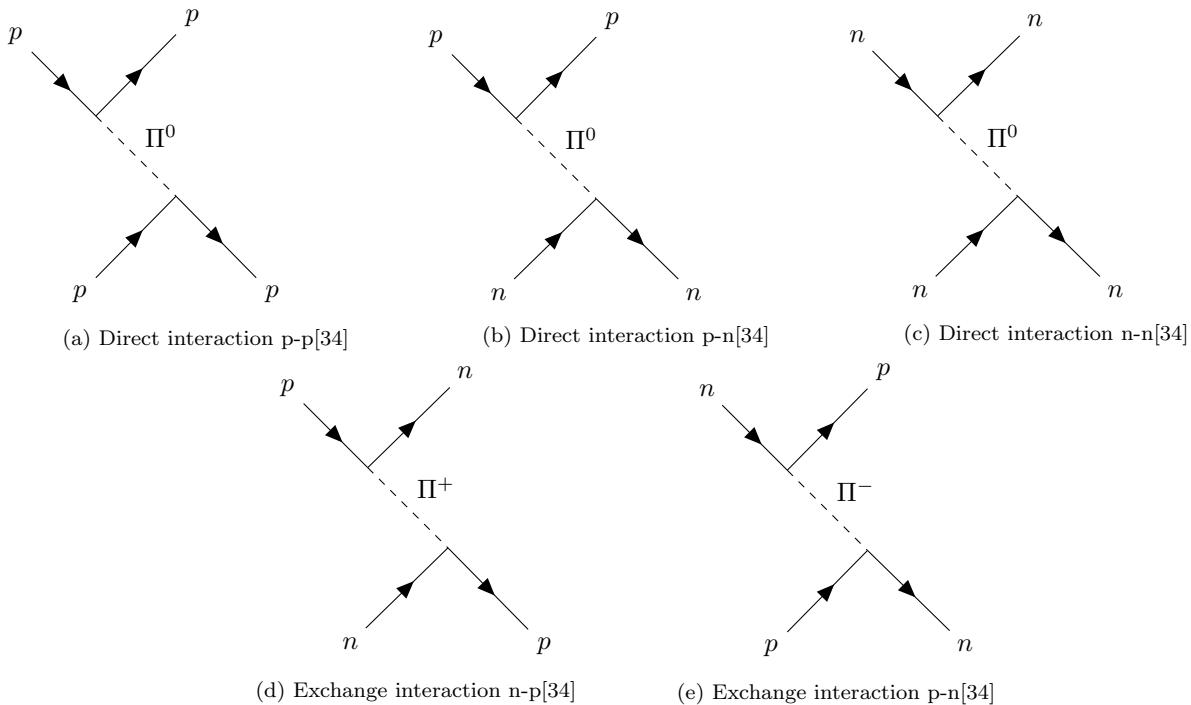


Figure 2.1: Proton (p) and neutron (n) interactions with the exchange of virtual pi mesons (Π^0 , Π^- , Π^+), carriers of the strong nuclear force, responsible for binding neutrons or protons to the nucleus during the activation of a material

For nuclear reaction and damage calculations, the interaction of neutrons with matter is represented by a cross section (section 4.4.3). Cross sections are used to calculate the probability of a certain interaction occurring and are dependent upon the target nucleus, how much of the material they pass through and the energy of the neutron.

The attenuation of neutron radiation is therefore a complicated process. A much larger thickness of shielding is required to stop neutrons, or at least bring them to thermal temperatures. Not only this, but the thermalized neutrons must then be captured. During this process various isotopes within the attenuating shielding will have been transmuted, possibly into radioactive isotopes. As a rough estimate if a certain amount of material will attenuate neutrons to 10% their original flux, then the same amount of material will reduce the flux further to 1%. This is however an over simplification and computer codes such as Monte Carlo N-Particle (MCNP) and TART more accurately model the transport of neutrons through materials.

2.1.4 Gamma Rays

The electromagnetic spectrum classifies photons based on their energy and/or source, but visible light, X-rays, gamma rays and so on are all the same elementary “particle”, the photon. A constraint on gamma rays is that they originate from the decay of an excited nucleus and are typically more energetic than X-rays. X-rays are photons created by electrons changing energy levels within an atom or by accelerating high energy charged particles.

During the early years of Quantum Mechanics, the relationship between the energy (E) and wavelength (λ) of a photon was discovered: the Planck-Einstein relation (eq. 2.1), where Planck's constant (h) is equal to $6.626 \times 10^{-34} m^2 kgs^{-1}$. The frequency (f) is the inverse of the wavelength.

$$E = hf \text{ where } f = \frac{1}{\lambda} \quad (2.1)$$

At very high energies (10s-100s MeV) gamma rays interact predominantly through pair production. At lower energy Compton scattering is the dominant interaction and it allows the gamma rays to lose momentum and energy in collisions with electrons. Finally, for energies below 10keV-100keV, gamma rays lose energy through the photoelectric effect.

Pair Production

The energy of photons in the database used for this work ranges from 1keV up to almost 10MeV. There are several ways high energy photons will interact with the atoms of a target material. The rest mass of an electron is 511keV. If the photon energy is greater than 1.02MeV, i.e. there is at least enough energy to create an electron and positron, there is a chance that the photon will create an electron-proton pair. The energy of the photon (hf) balanced by the electron mass (m_e), positron mass (m_p) and their kinetic energies (T_e , T_p) as shown in eq. 2.2.

$$hf = (m_e + m_p)c^2 + T_e + T_p \quad (2.2)$$

The creation conserves energy and mass, with excess energy carried away as the kinetic energy of the particle pair. The charge before the creation is zero, as the photon is neutral, and the charge after is also zero, with the -1 of the electron and +1 of the positron cancelling out. Angular momentum is also conserved; the photon is a spin 1 Boson and, as electrons and positrons are Leptons they have half integer spin, adding up to 1. Finally,

momentum is not conserved in a vacuum, and this is why pair production occurs in the coulomb field of a nucleus. The nucleus carries away excess momentum, fulfilling this conservation law (fig. 2.2).

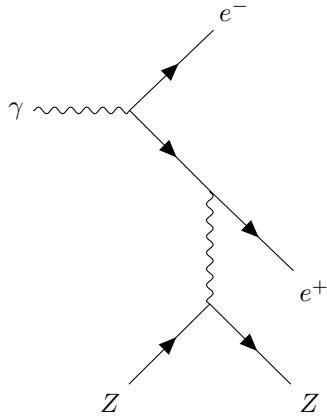


Figure 2.2: Pair production Feynman diagram, the process responsible for 511keV and 1.022MeV gamma lines[35]

Compton Scattering

An incident photon with enough energy may interact with the electron of an atom, and if it transfers enough kinetic energy it will eject the electron from the atom (fig. 2.3). A lower energy photon is also created that carries away the remainder of the energy, but also linear momentum, as both energy and momentum must be conserved.

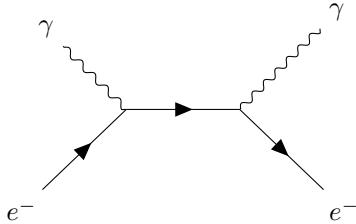


Figure 2.3: Compton Scattering Feynman diagram - responsible for attenuation of gamma rays through a material[36]

Photoelectric Effect

Photons of sufficient energy interact with and eject electrons from the surface of a metal (fig. 2.4). At lower energies, the photons have no effect on the electrons. Ejected electrons have a maximum kinetic energy (T_{max}) equal to the difference between the energy of the interacting photon and the workfunction (ϕ), $T_{max} = h\nu - \phi$.

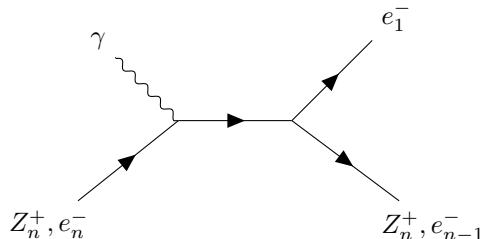


Figure 2.4: Photoelectric Effect Feynman diagram - absorption of gamma rays through the transfer of energy to an orbital electron[37]

Gamma Attenuation

Whilst an infinite amount of material is required to reduce the intensity of gammas to zero, they may be reduced by a significant amount with dense material. Lead, Uranium and other high density elements with a large number of electrons per unit volume are ideal. As highlighted in the previous paragraphs, gamma rays lose energy in material through Compton Scattering and the Photoelectric Effect.

$$I(\mu/\rho(E), t) = I_0 \exp\left(-\frac{\mu}{\rho}(E)\rho t\right) \quad (2.3)$$

The attenuation is dependent upon the thickness of the material (t) and its density (ρ), with a thicker and denser material reducing the intensity (I) of the gamma rays (eq. 2.3[38]). It is also dependent upon the initial intensity I_0 and the mass attenuation coefficients ($\frac{\mu}{\rho}$) that are in turn dependent upon the photon energy (E). A table of attenuation coefficients is available for each element[39].

2.2 Microscopic and Macroscopic Cross Sections

As discussed above there are a range of different ways in which matter interacts. The probability that a certain interaction will happen, which may be a function of the target, projectile and projectile energy, is described with the microscopic cross section. The total microscopic cross section, σ_{total} , is computed by summing all the individual reaction channels (eq. 2.4)

$$\sigma_{total} = \sigma_{elastic} + \sigma_{inelastic} + \sigma_{absorption} + \sigma_{fission} + \dots \quad (2.4)$$

The cross section has units of area and is measured in barns. When this area is multiplied by the target thickness and number density of the target, the fraction of projectiles that will react in this way is computed (see section 4.4.2).

Multiplying the microscopic cross sections by the number density (N_d) defines the macroscopic cross section (Σ_{total}) (eq. 2.5), whether it be for an individual reaction type or the sum of them all. This has units of atoms per meter and taking the inverse gives the mean free path, λ , of the projectile in the target material (eq. 2.6).

$$\Sigma_{total} = N_d \times \sigma_{total} \quad (2.5)$$

$$\lambda = \frac{1}{\Sigma_{total}} \quad (2.6)$$

The probabilities and associated libraries of a neutron, proton, deuteron and so on transmuting a target atom are discussed in more detail in section 4.4.3.

2.3 Quantifying Radiation and its Effects

The reason for attributing a value to a radiation source is usually because we want to quantify it and link that value to the effects it will have on materials, equipment and, most importantly, people. There are a number

ways of quantifying radiation, an associated number of different units as well as many other factors that affect how concerned we should be about a particular source.

Parameters most obvious to quantify radiation are the rate at which the particles are being created as well as the duration the equipment, material or person is exposed to that radiation.

Fluence (ϕ) is the measure of the total number of particles (N) passing through an area (A) (eq. 2.7). Whilst this is typically defined as the amount passing through a spherical area, the proton radiation that concerns this work is directional and it will be the planar fluence[40]. The fluence rate, flux ($\dot{\phi}$), is a measure of how many particles pass through a sphere (or plane where planar) per second.

$$\begin{aligned}\phi &= N/A \\ \dot{\phi} &= \frac{d\phi}{dt}\end{aligned}\tag{2.7}$$

Absorbed dose (D) is the amount of energy absorbed by a unit mass. This will be the sum of the fluence and energies of each individual particle and will depend upon the proportion of that energy deposited in the target. It is measured in the SI unit Gray although a historical unit is Rad, which is equal to 1 centigray.

$$H_T = \sum_R W_r \times D_{T,R}\tag{2.8}$$

Both the type of radiation and the type of tissue determine how much of an impact that radiation has per unit energy per unit mass. The equivalent dose (H_T) is calculated from the mean absorbed dose ($D_{T,R}$) weighted by the type of radiation (eq. 2.8[41]). The radiation weights (W_r) are determined by the Relative Biological Effectiveness (RBE) of the radiation type and may also be dependent on its energy. Typical values are given in table 2.2[42].

Radiation	Energy	W_r
Photons	all	1
Electrons	all	1
Neutrons	Less than 10 keV	5
Neutrons	10 keV to 100 keV	10
Neutrons	100 keV to 2 MeV	20
Neutrons	2 MeV to 20 MeV	10
Neutrons	More than 20 MeV	5

Table 2.2: Coarse radiation weighting factors[42]

The type of tissue is similarly accounted for in order to compute the effective dose (E) and requires tissue weighting factors (W_T) (eq. 2.9[42]).

$$E = \sum_T W_T \sum_R W_r \times D_{T,R}\tag{2.9}$$

Both the equivalent and effective dose are measured in the Si unit Sievert, and historically they have been measured in Rontgen Equivalent Man where the base unit is cGy rather than Gy.

2.4 Damage Cascades in Metals

When a metal is damaged by radiation microscopic defects are created and removed in a very short space of time. The result of many months or years of damage is complex and will be discussed in more detail in chapter 3.

Element	Cohesive Energy/eV
Aluminium	-3.36
Iron	-4.32
Palladium	-3.91
Platinum	-5.77
Zirconium	-6.32

Table 2.3: Cohesive Energies [43]

The energy of incoming neutrons, ions or fission fragments in a reactor and PKAs created by them are multiple orders of magnitude higher than the cohesive energies of atoms in the target material (table 2.3). Charged radiation will lose energy to electrons in the target and also directly with the nuclei of atoms, but neutrons will lose energy primarily in collisions with nuclei. When an atom is knocked out of place by a projectile (a neutron, ion or fission fragment), it becomes a PKA.

Projectile	Mean Recoil Energy
1 MeV Electrons	60eV
1 MeV protons	200eV
1 MeV heavy ions	5keV
1 MeV neutrons	35keV

Table 2.4: Average recoil energy of Nickel PKAs[44]

The PKAs create a damage cascade, and their mean recoil energy is dependent upon the energy and type of radiation. G. S. Was[44] gives examples of mean recoil energies for Ni atoms (table 2.4) and these show a range of three orders of magnitude between different forms of radiation of the same energy.

After each cascade, there will be a period of time where the material quenches, with a recombination of some interstitials and vacancies. The PKA creation and cascade thermal spike cover a time range of 10^{-18} s to 10^{-13} s, with the quenching phase lasting in the region of 10^{-11} s. However, the cumulative effect of radiation damage on a component within a reactor core may only become apparent after months or years.

Not all interstitials and vacancies recombine, but they may remain as separate interstitial vacancy pairs (Frenkel defects[45]). In a reactor the damage process repeats and constantly produces interstitials and vacancies. These aggregate into prismatic loops[46] and, as they cannot self-annihilate by glide[47], they remain in the material and continue to grow.

Damage cascades have been the subject of a number of studies and due to their short time scale and complexity, involving thousands to millions of atoms, are studied with computer simulations. Given the shorter stopping range of a charged atom, when compared with a neutron, PKAs with energies in the tens to hundreds of keV are now possible to model.

The immediate aftermath of interstitials created three damage cascades is shown in figure 2.5[48]. In this work by Stoller in the mid 1990s, it was possible to model such cascades with a Finnis-Sinclair potential. The supercell sizes ranged from 54,000 atoms for a 1.24keV PKA to 1,024,000 for a 61.3keV PKA. Given the improvement in

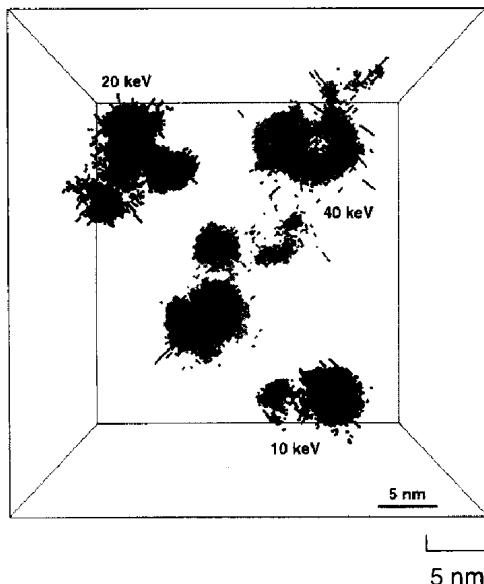


Figure 2.5: Interstitial formation in 10keV, 20keV and 40keV PKA displacement cascades and subcascades [48]. Only interstitials resulting from the three PKAs are shown in the simulation cell.

workstation computer power as well as petascale and exascale supercomputers today, the ability to capture the spatial aspect of a MD simulated damage cascade is assured.

By creating new potentials (chapter 7) there is the possibility of exploring RIS and the underlying defects and loops caused by radiation damage, in PGM doped austenitic stainless steels.

Chapter 3

Austenitic Steels in Nuclear Power

Austenitic stainless steels have played an important role in the nuclear industry. They have a good resistance to corrosion and behave well at moderately high temperatures, having a good resistance to creep. They have been used within the primary circuit and secondary circuit of power stations, as cladding for fuel pellets, in the steam dryers, pre-heater tubing, core structurals, primary piping and more. Ferritic steel is magnetic and has a BCC structure, whereas austenitic steel is non-magnetic with a FCC structure, with its high Nickel content stabilising this phase. An area of concern is the susceptibility of austenitic stainless steels to IGSCC.

3.1 Stainless Steel

3.1.1 Introduction

Stainless steel is a relatively new material, having first been developed and refined from the 1800s to the early 1900s, then being defined as a steel with at least 10.5% Cr in 1911. The addition of Cr to this level causes the formation of a passive protective layer of chromium oxide.

Fe is alloyed with Cr, Ni and C in varying quantities to make Stainless Steel. Depending on the application, other elements, such as Mo, may be added to enhance the properties of the steel.

3.1.2 Grades of Stainless Steel

While the criteria that qualifies an alloy as Stainless Steel is containing Fe, C and at least 10.5% Cr, there are many grades that have a variety of properties and atomic structures. The composition influences the structure and other properties such as resistance to corrosion, yield strength, creep resistance and whether the steel is magnetic or not.

Both Fe and Cr are BCC at standard temperatures and pressures, but adding austenite stabilisers such as Ni, Mn or C, the structure of the steel can be altered from BCC to FCC. Balancing the proportion of these elements changes the phase of the steel, and this may be represented as a Schaeffler diagram (fig. 3.1[49]) or a ternary phase diagram (fig. 3.2[50]).

Ferritic Stainless Steel

At room temperatures and pressures, Fe exists in the alpha phase, which is a BCC crystal. It is energetically favourable for the magnetic moments of the Fe atoms to align with one another ferromagnetically. Ferritic

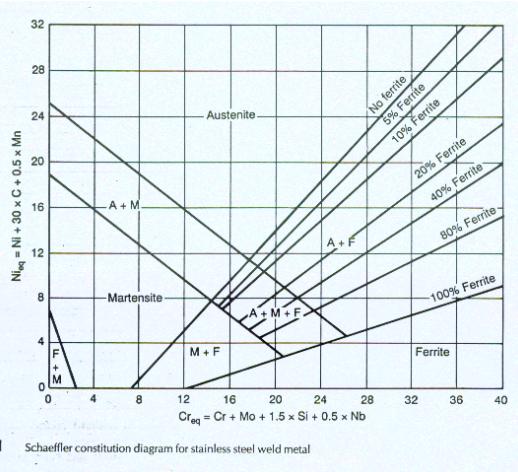


Figure 3.1: Steel Cr-Ni Schaeffler Diagram[49]

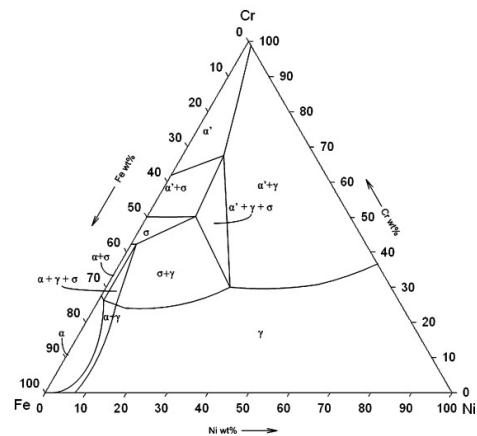


Figure 3.2: Iron-Chromium-Nickel Phase Diagram[50]

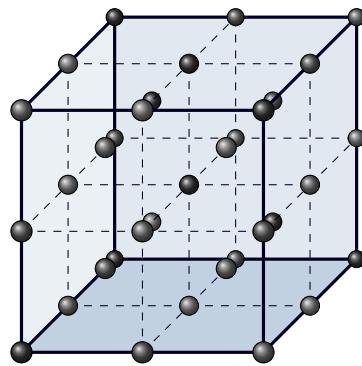


Figure 3.3: BCC structure of ferritic stainless steel - a 2x2x2 supercell[51]

stainless steels also have the natural alpha phase crystal structure of pure Fe at room temperature which is BCC (fig 3.3). These steels are magnetic and may be hardened by cold working. They are less corrosion resistant than austenitic stainless steels. Two common examples of this grade of steel are American Society of Engineers codes 405 and 430.

Austenitic Stainless Steel

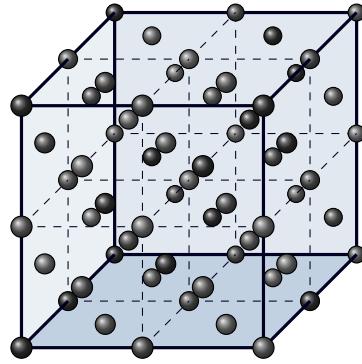


Figure 3.4: FCC structure of austenitic stainless steel - a 2x2x2 supercell[51]

Austenite is a FCC allotrope of Fe (fig 3.4), and austenitic stainless steels are useful in many applications, including a structural material for nuclear plant components, due to their resistance to corrosion. In addition to 10.5 wt% or more Cr they require an austenite stabilising element to be added.

Two examples of such steels are ASME codes 304 and 316. Both have a high Cr content, in the region of

18-20%, which is in excess of the minimum passive film requirement of around 10-11%. The natural structure of such an Fe-Cr alloy would be BCC, however 304 and 316 Steels contain Ni (approximately 8% and 10% respectively) which stabilises the austenite phase and is responsible for the FCC structure of the steel. The 316 grade contains a minimum of 2% Mo to improve its resistance to corrosion.

Ferritic and martensitic have a better resistance to thermal or swelling shock than austenitic [52], but the corrosion resistance of austenitic steels is a deciding factor when choosing a steel for an application where corrosion resistance is important.

Martensitic Stainless Steel

Martensitic stainless steels have a FCC structure at high temperature, but when heat treated take on a BCC structure. This allows martensitic, unlike austenitic and ferritic, to be hardened by heat treatment. These steels are magnetic, they contain more than 10.5% Cr but have a much lower Ni than austenitic grades, if any. They are corrosion resistant, but the corrosion resistance of austenitic steel is typically better. Two examples of such steels are ASME codes 410 and 431.

3.2 Austenitic Steels in Nuclear Reactors

3.2.1 The Use of Austenitic Steels in Reactors

Austenitic stainless steels are known to have better corrosion resistance when compared to their ferritic and martensitic counterparts. Due to this and their excellent strength, austenitic steels have been widely used in nuclear reactors. The cheaper to manufacture 304SS and more corrosion resistant 316SS have been particularly popular choices of steel. There are drawbacks when compared to ferritic and martensitic steels, and two in particular are IGSCC and swelling.

3.2.2 Atom Damage Cascades

Fission fragments carry away the majority of the energy released during fission, with an approximate energy of 170MeV per ^{235}U fission event. The fragments are large and have a highly charged positive nucleus that stops rapidly due to the Coulomb interaction with the nuclei of surrounding atoms. Electrons released by beta decay travel a greater distance; they lose energy by interacting with electrons, but may also knock atoms out of place. Neutrons travel further still and lose energy by interacting directly with the nuclei of the surrounding material.

When an incoming projectile hits an atom in the target material, this atom becomes the primary knock-on atom (PKA). This PKA is knocked out of its place and loses energy by colliding with other atoms and through electronic stopping. If its energy is high enough it will cause a damage cascade through the material. Atoms knocked out of their regular position in the crystal lattice leave vacancies behind. The displaced atoms either recombine with vacancies, become interstitial atoms or diffuse to defect sinks.

The damage event has four main phases over a short period of time[53]:

- PKA creation (10^{-18}s)
- cascade and thermal spike (10^{-13}s)
- quench (10^{-11}s)
- annealing and defect migration (10^{-9}s)

These individual damage events happen over a very small amount of time, but a number of complex problems arise as a result of this damage over a much longer time period. The damage to components can take many years and high damage doses to manifest on a macroscopic scale.

3.2.3 Damage Rates

The safety of a Nuclear reactor is the primary concern; it must generate power, but it must be safe. It must also be cost efficient. Gen II PWR fuel assemblies were expected to withstand several DPA[54]. In BWRs the materials are designed to be irradiated by a total fluence of $10^{22} ncm^{-2}$, experiencing approximately 7 DPA over the lifetime of the components[55]. Components in PWRs are irradiated by up to a total fluence of $10^{23} ncm^{-2}$, and are expected to operate up to an irradiation dose of 70 DPA[55]. To remain cost efficient, components in Gen IV reactors will be expected to operate safely up to even higher doses or irradiation damage (fig 3.5). Components in sodium-cooled fast reactors will be expected to withstand up to 200 DPA over their lifetime to meet the requirements of cost effectiveness and durability[54].

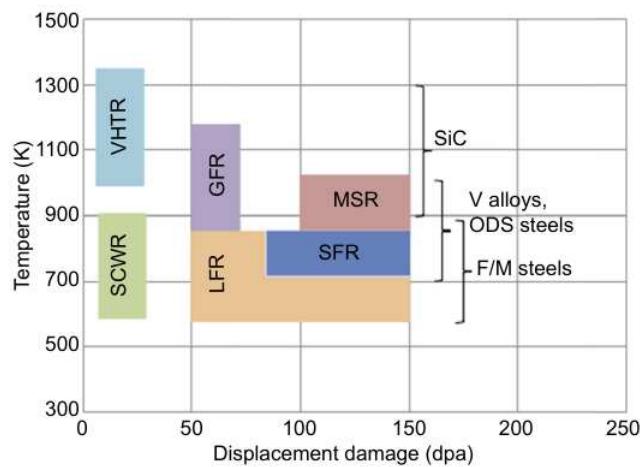


Figure 3.5: Expected DPA during component life time and operating temperatures[54]

Neutrons released during the fission of $^{235}_{92}U$ range from 0 to 14 MeV (fig 1.7). The energy transferred to a target atom (E_{tr}) depends on the neutron energy (E_n), the recoil angle of the neutron (θ) in the lab frame and the mass of the target nucleus (m_a). With the neutron mass (m_n) the energy transferred may be computed using eq 3.1[40].

$$E_{tr} = E_n \frac{4m_a m_n}{(m_a + m_n)^2} \cos^2 \theta \quad (3.1)$$

In a collision, the amount of energy transferred to a target atom will depend on both the scattering angle of neutron recoiling from the target and the mass of the target nucleus (fig. 3.6). If the neutron does not change direction, a small amount of energy will be transferred, but if it bounces back at 180 degrees it will transfer the maximum amount of energy (fig. 3.7). A neutron will lose more energy per collision with light atoms (hydrogen, helium, carbon), but much less energy per collision with larger atoms (iron, molybdenum, lead).

The energy of Fe PKAs as a result of neutron irradiation has been calculated for an Fe target[27] (fig. 1.8), and those created by 1MeV neutrons may have an energy up to 100keV. As nuclear reactions are also possible between the neutrons and the target nuclei, there is an inelastic (n, n) ^{56}Fe channel as well as other elastic scattering channels where the post collision particles are different to the pre collision particles (fig. 3.8).

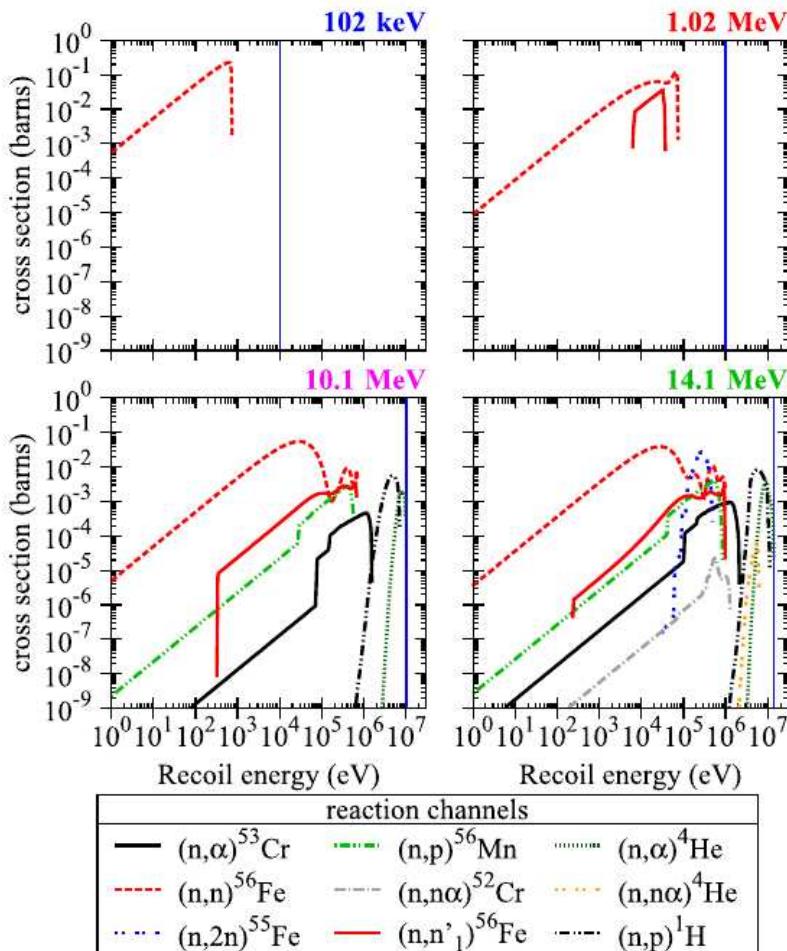
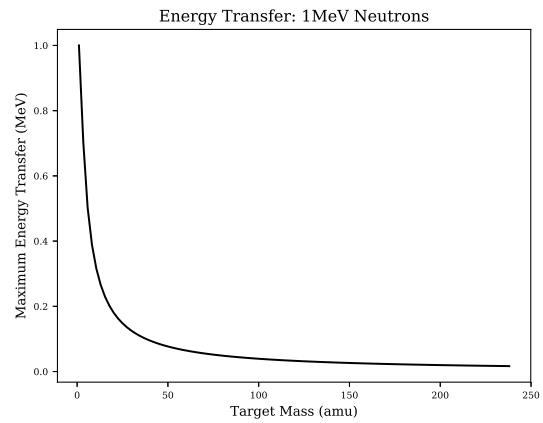
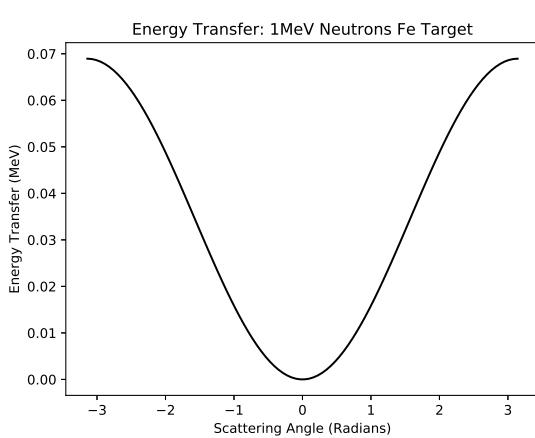


Figure 3.8: Recoil energies for neutrons at 102keV, 1.02MeV, 10.1MeV and 14.1MeV[27]

3.2.4 Swelling

Swelling results from damage to the crystalline structure of the metal and the temperature range for such swelling in steels is primarily from 670K to 870K. Radiation damage creates a constant supply of Frenkel defects and these take up more volume than the relaxed undamaged crystal. The vacancies and interstitials may group into prismatic loops of either vacancies or interstitials as these are thermodynamically more stable than the defects being dispersed throughout the crystal[56]. The creation of vacancy and interstitial loops should balance each other out in terms of the change to the volume, but when the vacancy loops meet in three dimensions they form cavities and these do not decrease the volume of the material[46]. The interstitials are the cause for the change in volume, and so the material swells.

Materials have been known to swell to a 100% increase in their original volume at these intermediate temperatures under irradiation[57]. Swelling is sensitive to the total damage amount, the temperature and composition of the alloy. An increase in Ni decreases swelling as does the addition of Zr. Small amounts of P and Mo (0.02% and 0.5-1.0% respectively) cause the swelling of the alloy to increase, however higher concentrations of either cause swelling to decrease[58].

As a component swells its properties change. By definition the volume changes, but so does the elastic moduli. Not only this, but the swollen component may stress itself and surrounding components, and this may lead to SCC. Due to the formation of voids and interstitial dislocation networks, stresses are also cause within the material on the microscopic scale.

The swelling of components is a problem experienced in past and current reactors. In the EBR-II structural 304SS was irradiated to 20DPA. At a temperature of approximately 650K the steel's volume increased by 2% due to swelling[59]. With components in future reactors expected to withstand ten times this damage, swelling is a concern. However, increased temperatures and a careful balance in the composition of future alloys will help to reduce this.

3.2.5 Radiation Induced Segregation

The diffusion of atoms within an alloy has an impact on the characteristics of the metal [60]. When radiation causes point defects, these interstitials also diffuse parallel to thermal solute diffusion. At low temperatures, the atoms are unable to diffuse at an appreciable rate; the mobility of vacancies are low[44] and there are an excess of vacancies due to radiation damage, and this leads to recombination of defects. At high temperatures, there is a higher concentration of thermal defects[55]. This increases the defect recombination rate and reduces migration of defects to sinks, such as grain boundaries.

The melting temperature range of 304 and 316 stainless steel are approximately 1695-1722K and 1644-1672K respectively. Gen II reactors, such as Sizewell B, have an operating temperature of several hundred degrees centigrade. The inlet temperature for the Sizewell B PWR reactor is 566K[61] and the outlet temperature is 597K[61]. Operating at approximately 35% the melting point of the steel this is, unfortunately, a good temperature for radiation induced segregation (RIS) to occur.

The Kirkendall effect (KE) concerns the diffusion of one material into another and vice versa at an interface. In the original experiment performed by Kirkendall and Smigelskas, brass (Cu and Zn) was sandwiched between Cu and left at a temperature of over 1050K for almost two months. Using Mo as a marker, it was discovered that Zn diffused out of the brass faster than the Cu diffused into the brass. The flux of atoms results in a flux of defects into and a shrink in volume of the brass[62].

The inverse Kirkendall effect (IKE) is driven by an external force, such as irradiation. Irradiation of the material causes a flux of defects and, inverse to the KE, this causes a flux of atoms. For an austenitic stainless steel the diffusion coefficients in order of size are $D_{Cr} > D_{Fe} > D_{Ni}$ [63], so Cr will diffuse away from grain boundaries

and other sinks, followed by Fe. As Ni is slowest to diffuse, this will be enriched. An additional mechanism that is thought to contribute to RIS is interstitial flux where undersized atoms as interstitials are preferential[63].

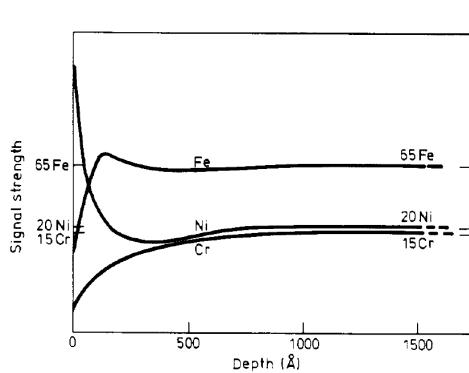


Figure 3.9: Depletion of cr, fe and enrichment of ni at the surface[64]

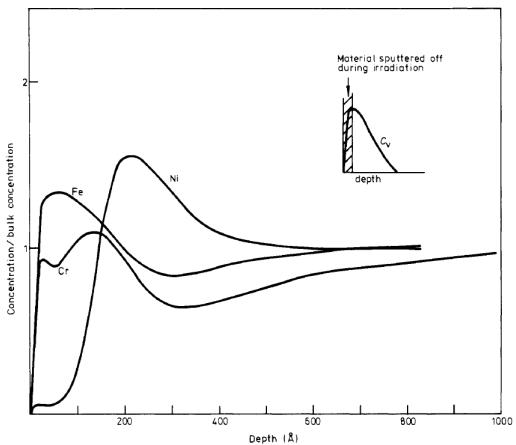


Figure 3.10: Concentration profile after 46DPA irradiation with 75 keV Ni ions[65]

Experimental work by Johnston et al irradiated steel, containing 15% Cr and 20% Ni, with 4MeV Ni ions. It was irradiated to a damage dose of 8DPA at 950K. Some material will have been sputtered from the surface, but in the remaining material Cr, and Fe, were depleted. Ni, however, was enriched (fig. 3.9).

In work by Marwick a similar alloy with 14% Cr and 15% Ni was irradiated to a much higher dose of 46 DPA with lower energy 75KeV Ni ions. In this instance the damage occurs in a much thinner layer of the surface that was approximately 30nm thick (fig. 3.10). Conversely to Johnston et al, in this thin layer there is a depletion of Ni and an enrichment of Cr in the first 20nm or so. However, between approximately 25nm and 60nm there is a similar depletion of Fe and Cr as well as an enrichment of Ni.

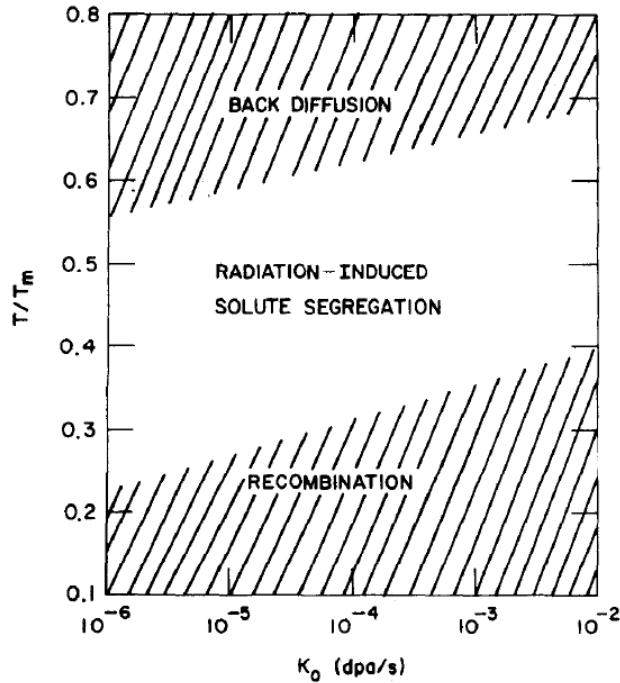


Figure 3.11: Temperature and DPA dependence of RIS[66]

The underlying mechanisms of RIS are dependent on both the temperature (relative to the melting point of the material) and the amount of damage (fig. 3.11). The result of RIS is a breakdown in the passive protection of

Cr, which leads to IGSCC under the correct conditions.

3.3 Corrosion Resistance of Austenitic Stainless Steels

3.3.1 Passive Film Protection of Chromium

The addition of Cr improves the resistance of stainless steel to corrosion by orders of magnitude. A very thin passive oxide layer, 20-30 angstroms in width, forms when the surface is exposed to an environment containing O. The protective layer is self repairing and as such, if the surface is damaged, the oxide layer reforms in the presence of O.

When the steel is in an environment containing O, electrons within the metal tunnel through the surface. As an oxidizing agent, the nearby O atoms readily accept the electrons. A strong electric field forms between positive ions within the metal and the negatively charged O atoms [67].

Fe at the surface of the forming oxide layer is preferentially dissolved away over Cr [68] and Cr ions within the oxide layer have a lower mobility than Fe ions [69]. Ni remains in place within the alloy, and this leads to an enrichment of chromium oxide in the passive layer. Once the layer is thick enough to reduce the electric field across it, the formation of the layer stops. As mentioned previously, this is at a layer thickness of 2-3nm for stainless steel.

The Cr_2O_3 layer may be formed by heating the steel to 770K[70]. However, at higher temperatures, the steel begins to lose its protective layer due to sensitization.

3.3.2 Sensitization and Passive Film Removal

Steel by definition is Fe alloyed with varying small percentages of C. The addition of C changes the property of the alloy, and one example of this is an increase in hardness over pure Fe. At elevated temperatures, 670K to 1020K, the added Cr within the steel forms precipitates of Fe-Cr carbides $(\text{FeCr})_{23}\text{C}_6$ at the grain boundaries, reducing the percentage of Cr at the grain boundary and removing the layer of passive protection (fig. 3.12 & 3.13).

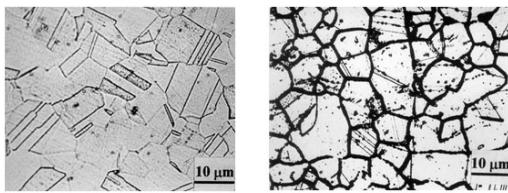


Figure 3.12: Formation of chromium precipitates at the grain boundary [71]

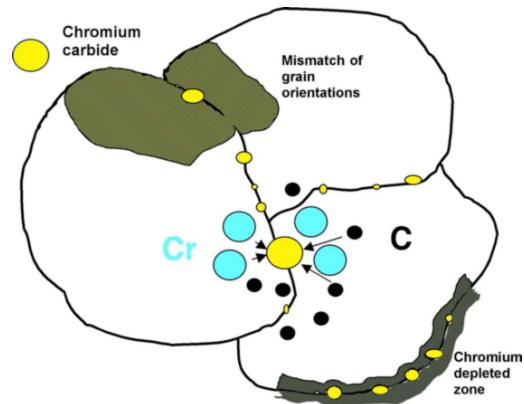


Figure 3.13: Sensitization of an alloy: chromium carbide precipitation [72]

The sensitization of the steel can be reversed. By heating the steel further, to approximately 1,320K to 1,420K, the steel is solution annealed dissolving the carbides back into the steel.

3.3.3 Addition of Molybdenum

A common austenitic stainless steel is 304SS and this relies heavily on the passive film due to a high content of Cr, ranging from 17.5% to 20% for grades 304, 304L and 304H. A similar stainless steel, 316, has slightly less Cr, slightly more Ni and 2-3% Mo, an element not present in 304 stainless steel.

Pure Mo has a BCC crystal structure, and it is a ferrite former. However, high proportions of austenite former such as Ni and N force 316 stainless steel to keep its FCC structure despite the addition of Mo. Adding it to austenitic steels improves their strength and resistance to creep at high temperatures.

Steels such as 316SS are known to have better corrosion resistance in chloride rich environments. Where at least 2% Mo has been added it improves resistance to pitting and crevice corrosion resistance[73]. The mechanism by which Mo helps to protect against corrosion is still unclear. It is possible the addition helps to reduce the breakdown of the passive film protecting the steel, but it may also be the case that Mo promotes the repair of the passive film [74].

3.4 Irradiation Assisted Stress Corrosion Cracking

Damage due to irradiation was first identified in high stress stainless steel components in a reactor, such as bolts, springs, and fuel elements[75][76], and later being found in lower stress austenitic stainless steel components[76].

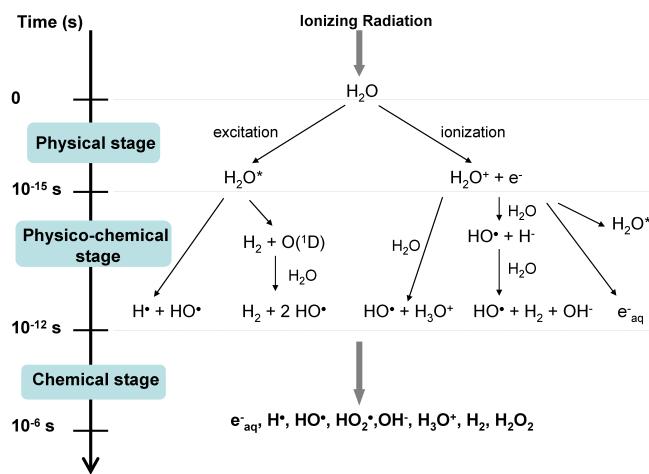


Figure 3.14: Changing water chemistry: radiolysis of water[77]

Irradiation of water causes the radiolysis of water, changing the chemistry of water and creating hydrogen, radicals and other ions that increase the corrosiveness of the environment (fig. 3.14). The radiation also damages the material directly, changing its properties. A form of IASCC to which high nickel alloys, including austenitic steels, are particularly susceptible to is inter granular stress corrosion cracking.

3.5 Inter Granular Stress Corrosion Cracking

Stress corrosion cracking may be trans granular (through the grains) or inter granular (fig. 3.15), at the grain boundary. IGSCC is a particularly prominent failure mechanism for austenitic stainless steels. For IGSCC to occur, there must be a material that is susceptible to this form of cracking, an environment that is corrosive to the material as well as stress (fig. 3.16).

Stress may come from a high pressure environment, residual stresses due to welding and swelling. In a nuclear reactor the swelling on a macroscopic scale is a result of damage caused by neutrons on the atomic scale as they

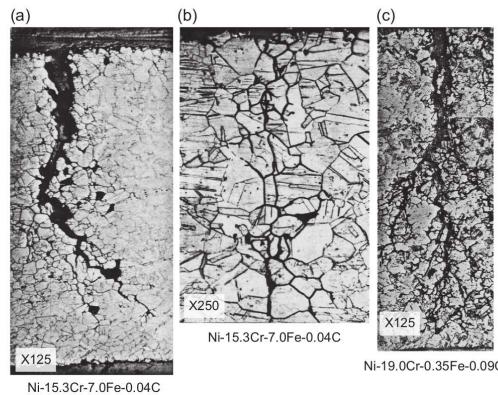


Figure 3.15: Inter Granular Stress Corrosion Cracking in Nickel Alloy[78]

pass through the steel creating defects. If Cr is depleted at the grain boundary, protection to corrosion due to the passive layer is lost. This, coupled with the knowledge that austenitic stainless steels are susceptible to IGSCC, completes the three requirements and, over time, components of this material in these conditions will eventually fail to IGSCC.

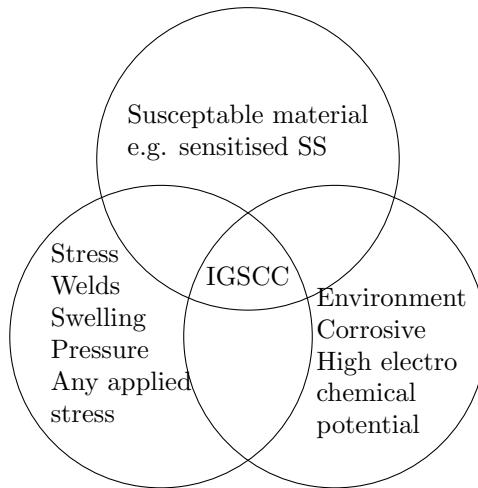


Figure 3.16: Three requirements for IGSCC to occur

IGSCC has not only been a defect in stainless steel. High Ni content alloys, such as Alloy 600 (Inconel: Ni-72, Cr-17, Fe-10), have been known to suffer from IGSCC since the very early days of nuclear energy, in particular with the prototype S1W reactor, prototype for the first nuclear powered submarine, the USS Nautilus.

After the war, there was a drive to develop a nuclear powered navy for the US. There are obvious benefits in replacing conventional power in vessels with nuclear, in particular for submarines. This was pushed by Admiral Rickover and the route to building the USS Nautilus began. A number of available Fe-Cr-Ni alloys were considered for the construction of the reactor (fig. 3.17).

Prototypes developed for the USS Nautilus experienced stress corrosion cracking in the Inconel Alloy 600 components. H. Coriou replicated this damage to Inconel in the laboratory, holding a sample at 620K in deoxygenated water for 3 months[78]. Further studies showed that IGSCC became a particular issue to alloys with a high Ni content in pure water, commencing with a content of approximately 70% Ni. In a weak salt water solution (0.1% NaCl) IGSCC occurred as with the same alloy in pure water, but TGSCC occurred in similar alloys with a low Ni content (less than 15%) (fig. 3.18).

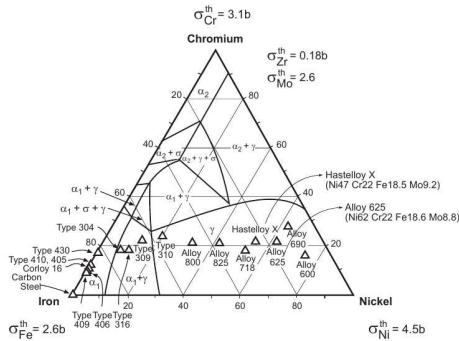


Figure 3.17: Alloy choices for early LWRs[79]

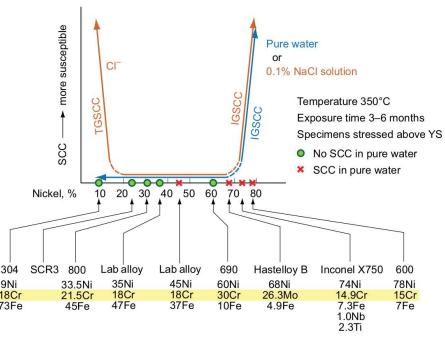


Figure 3.18: Nickel, IGSCC and TGSCC[80]

3.5.1 IGSCC in Light Water Reactors

In LWRs a major factor in corrosion of components is the water chemistry and electrochemical corrosion potential that they operate in. Higher levels of oxygen dissolved in the water leads to an increase in the corrosion potential[57]. Other factors within LWRs include depletion of Cr due to either heat treatment, welding or irradiation.

The primary circuit of a BWR includes components such as a reactor pressure vessel, piping that leads out of the containment building, fuel and fuel assembly. 300 series steels, including 304 and 316, are used within the primary circuit, including the piping and control rod absorbers. In BWRs the purity of the water has been addressed, and the addition of hydrogen to the water has reduced cracking of components[78].

Higher nickel content steels such as the 600 series alloys are used in PWRs, and components that are made from these alloys include steam generators and pressure vessels. In the AP-1000 design, the control rod drive mechanism at the reactor coolant pressure boundary are made from 304, 304L, 304LN, 316, 316L and 316LN steel. Higher nickel content 690 is used for penetration into the pressure vessel[81]. As illustrated by Coriou's work in the 1950s and 1960s, higher nickel content steels are susceptible to stress corrosion cracking.

3.5.2 IGSCC and Advanced Gas-cooled Reactors

A major drawback of the first generation Magnox reactors was their relatively low operating temperature of 630K. Carnot's theorem shows that the maximum amount of energy from an engine is dependent on the difference between the hot and cold reservoirs (section 1.2.3). The ambient temperature of the power station will vary a small amount with the seasons. The practical way to improve the maximum possible efficiency is to increase the engine (reactor) temperature.

AGR reactors were designed to run at much higher temperatures of 920K. By increasing the temperature, the maximum possible thermal efficiency was increased from just over 50 percent to almost 70 percent. To withstand higher temperatures, the magnesium oxide cladding used in the earlier Magnox reactors was replaced with stainless steel. To counteract the higher neutron absorption cross section of the cladding, the fuel was enriched up to 3.5% U²³⁵.

The cladding holds the fuel elements together and at the required location inside the reactor whilst it is consumed in the reactor. The cladding also performs several functions once the fuel has been spent. It holds the fuel elements together and acts as a primary containment for the spent fuel[82].

During its lifetime as a fuel element within the reactor, the cladding has been heated and irradiated. By the process of RIS the Cr at the grain boundary can drop to 10% concentration[83]. The C rich environment within the reactor, due to the CO₂ coolant, provides yet another mechanism to deplete Cr at the grain boundary by the formation of Fe-Cr carbides (FeCr)₂₃C₆.

This loss of protection at the grain boundary, stress due to the role of the component within the reactor, the changing environment due to radiation damage and the material's susceptibility lead to IGS SCC.

3.6 Austenitic Stainless Steels in Gen III+ and Gen IV Reactors

The AP1000 is an advanced PWR. As a Westinghouse reactor, the fuel cladding will be their trademarked Zirlo alloy. Inconel will be used for the steam generator and heat exchanger with carbon steel used for the construction of the pressure vessel. The control rod absorbers, however, will be constructed from 304SS[84].

Areva designed the EPR and this will use 316SS as the fuel cladding material. It will also use the slightly less corrosion resistant 304SS (forged) in parts of the control rod drive mechanisms[85].

The perforated upper core plate, between the reactor core and the inlet/outlet/control rod mechanisms, is also made from austenitic stainless steel. The pressuriser is constructed from ferritic steel, but its internals are clad with austenitic stainless steel to protect the structure from the coolant.

Gen IV designs also include the use of austenitic stainless steels. Experimental fast reactors in many countries, including the US, UK, Japan, France, India and China have used either 304SS, 316SS or both in their construction.

Austenitic steels are more corrosion resistant than ferritic or martensitic steels. They do expand more as a result of being irradiated, but they are stronger at higher temperatures. Many of the Gen IV reactors operate at much higher temperatures than existing reactors and the components will be expected to resist higher doses of radiation damage throughout their lifetime.

GFRs are expected to use austenitic steels. Due to their chemical compatibility with sodium they are also expected to be used in SFRs. As well as being used in the reactor core, these steels will also be used outside the core. For example, SFRs in France will use austenitic steels in the secondary circuit, primary pump, internal heat exchanger and more[86].

3.7 The Addition of PGMs to Stainless Steel

Water purity is a concern for light water reactors. The fewer contaminants there are to begin with, the better. Even so, the radiolysis of the purest water will still create a steady supply of corrosive molecules (fig. 3.14). Adding hydrogen to the water reduces the electrochemical corrosion potential of the water, but if too much is added it will combine with the radioactive nitrogen-16 created by the $^{16}O(n, p)^{16}N$ reaction to form $^{16}NH_3$ [87].

Cathodic modification is a technique used to improve corrosion resistance, and one way to do this for stainless steel is to add small amounts of platinum group metal (PGM)s[88]. By adding such metals, the anodic reaction may be reduced. PGMs also act as a catalyst in the reduction and removal of O_2 and H_2O_2 from the water[87].

The increased corrosion resistance is dependant upon the amount of PGM added, but a higher percentage does not necessarily mean better protection. This is important for two reasons, to find the optimum amount for the sake of corrosion resistance, and to keep the price down as the additions are so expensive, even in such small amounts.

It was found that an addition of 0.15% Ru (out of a selection of 0%, 0.05%, 0.10%, 0.15% and 0.20%) was optimum for both Fe30Cr and Fe40Cr steel (fig. 3.19)[88]. Other PGMs may be used for cathodic modification, including Pd, Ir and Pt, but the cost of the metal must be weighed against the possible gains.

Connolly et al added Ru and Pd to 304SS samples that were sensitized at 920K for 24 hours[89]. After sensitization, the steel was analysed by TEM and Cr-rich carbides were found at grain boundaries. Without the passive protection due to Cr, the samples were exposed to polythionic acid to promote SCC. The Ru doped

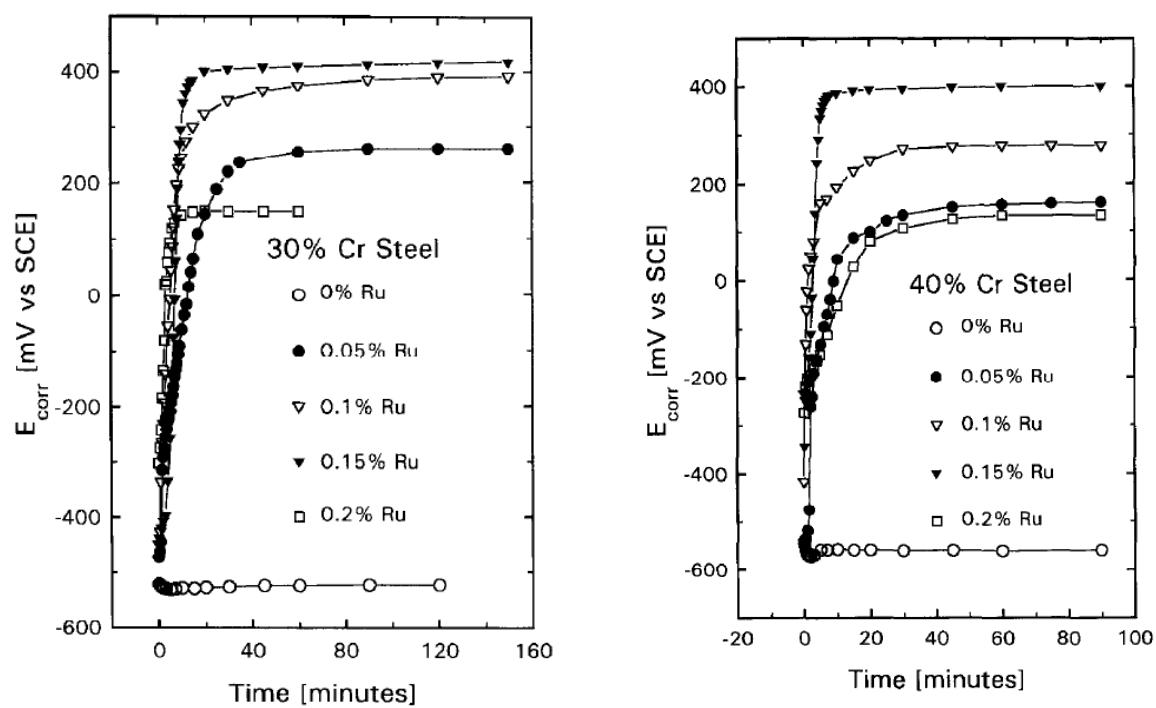


Figure 3.19: Addition of Ru to Fe30Cr and Fe40Cr steels - open current corrosion potential variation with time in 10% sulphuric acid[88]

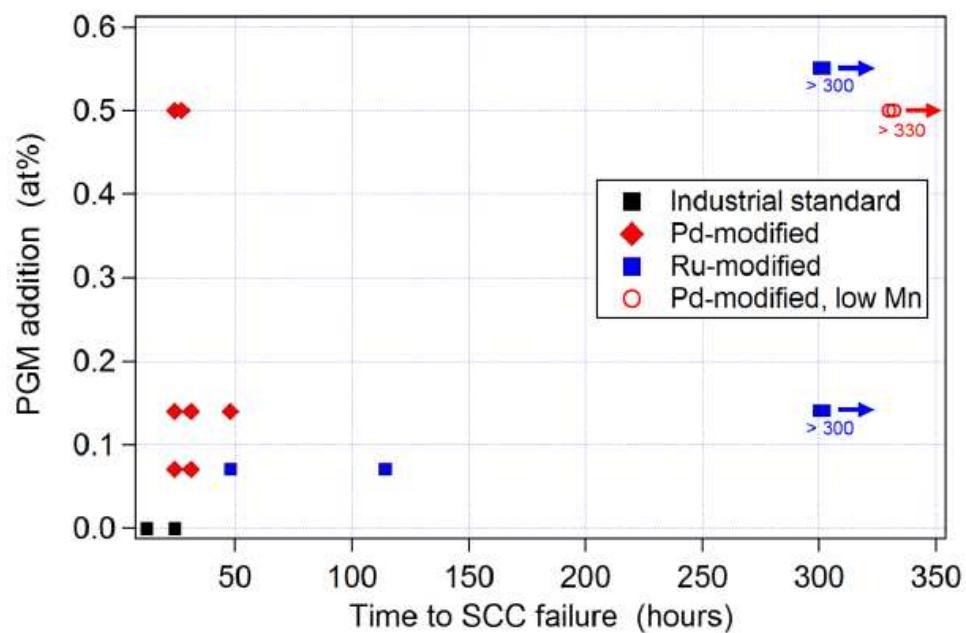


Figure 3.20: Failure times for 304SS, Pd doped 304SS (with and without Mn) and Ru doped 304SS in polythionic acid at ambient temperature[89]

steels performed well, but the Pd doped steels showed no improvement over regular 304SS. Further analysis of the samples by TEM reveals that the low percentage of Mn in the steel allows the formation of Pd-Mn precipitates at the grain boundary[89]. These precipitates form during the sensitization procedure and thus negate the protection by cathodic modification at the surface of the metal by removing the Pd.

With a low Mn 304SS (less than 0.05% Mn), varied amounts of Pd have been tested. With the reduction in Mn the corrosion resistance improves greatly over regular 304SS. Palladium remains at the surface, as the amount of Mn is too low for appreciable amounts of Pd to be lost to precipitate formation during sensitization. No failures have been observed with either the Ru or Pd (low Mn) samples for 330 hours exposure to polythionic acid at ambient temperatures (fig. 3.20) [89].

The surface of the metal does not need to have a full coating of PGM but it should be equally spaced across the surface. Oxidant in the layer at the surface will be consumed at the locations of PGMs and the change in concentration at those points will naturally cause diffusion of more oxidant to those locations[87].

The PGMs may be coated on the surface of the metal or alloyed to the metal. The first method is the more cost efficient but if a crack does form, or if a layer is removed, it will expose the regular alloy underneath. If the PGM is alloyed, there will always be protection, unless a mechanism is removing the PGM from the surface. This could be through precipitation as a carbide, or it could be due to radiation causing the metals to segregate.

Reaction	Product Halflife
$^{54}Cr(n, \gamma)^{55}Cr$	3.5 mins
$^{64}Ni(n, \gamma)^{65}Ni$	2.5 hrs
$^{55}Mn(n, \gamma)^{55}Mn$	2.6 hrs
$^{104}Ru(n, \gamma)^{105}Ru$	4.4 hrs
$^{108}Pd(n, \gamma)^{109}Pd$	13.7 hrs
$^{196}Pt(n, \gamma)^{197}Pt$	19.9 hrs

Table 3.1: Radioactive products and their half lives for PGM doped 304SS

Adding elements to metals, even in small amounts, may be problematic as certain elements have high neutron reaction cross sections and transmute into radioactive isotopes. ^{55}Cr is a concern for all neutron irradiated stainless steels, but with a very short half life it quickly cools and becomes less of a concern, with ^{65}Ni becoming the dominant source of activity after 10 hours. Adding Mn to steel that is in a reactor is known to result in an increase in activity due to the $^{55}Mn(n, \gamma)^{56}Mn$ reaction.

The addition of PGMs alters the activity of the steel following neutron irradiation and with just a 1% addition there is an increase in activity comparable to that of the 10% Ni transmuted to ^{65}Ni . This is due to reactions that create ^{105}Ru , ^{109}Pd and ^{197}Pt for Ru, Pd and Pt respectively (table 3.1).

With such small additions required for cathodic modification and the short half lives of the radioactive isotopes, there isn't a concern due to neutron activation over regular 304SS, and much less of a concern than that of 304SS containing Mn.

Chapter 4

Isotope Activation and Radioactive Decay

Using a proton source is an attractive alternative to neutron sources as they can generate similar damage doses at a much smaller cost. Isotopes in a material may be transmuted through either neutron or ion irradiation, so this remains a concern. The probability of transmutation is modelled using the OMP and this data is available in ENDF.

Radioactive isotopes take time to decay to safe levels, and the chain they follow may be complex. Bateman developed an equation to model this. This equation is developed further in chapter 6 resulting in a code called Activity. This couples the new equation with cross section data and ion trajectories from TRIM and SRIM.

In this work radiation activation will be modelled and potentials developed to model radiation damage. This chapter outlines experimental sources of ions and neutrons followed by a discussion of activation and the ion transport codes SRIM and TRIM.

4.1 Ion Sources

4.1.1 LINAC

Since the development of the first linear accelerator (LINAC) in the 1940s, their modern day versions have become some of the most powerful accelerators in the world. The longest LINAC, Stanford Linear Accelerator Center, is 3.2km in length and it accelerates electrons and positrons at energies of up to 50GeV. Several LINACs for protons include the 800MeV LINAC component of the ISIS neutron source in Oxfordshire, and the 800MeV LINAC used by the Spallation Neutron Source at Oak Ridge National Laboratory.

The accelerator is constructed of several tubes, connected alternately to opposite terminals of a high frequency alternating current supply. As protons enter the first tube, a negative voltage is applied. As the protons reach the gap between the first and second tube, the polarity is reversed. The positive charge that is now applied to the first tube pushes the protons forward as the negative charge on the second tube pulls the protons forward. This process is repeated along the length of the accelerator, with the sections increasing in length due to the increase in velocity of the protons.

4.1.2 Cyclotron

Cyclotrons are reasonably compact and cost effective. The largest current cyclotron, TRI-University Meson Facility, is located in Canada and is able to output protons with energies over 500MeV. It is relatively large, weighing 170 tons. There are approximately 350 cyclotrons[90] around the world today.

The University of Birmingham cyclotron is more compact. The protons it accelerates are in the 8-40MeV range and the projectiles are not just restricted to protons (table 4.1).

Projectile	Energy (MeV)	Maximum Current (micro A)	Projectiles $cm^{-2}s^{-1}$
proton	8-40	60	5.85×10^{14}
deuteron	8-40	30	2.93×10^{14}
He^{2+}	8-53	30	2.93×10^{14}

Table 4.1: University of Birmingham Cyclotron Ion Beams. The projectiles per area is based on a 0.8cmx0.8cm beam aperture.

The moderate size, energy range, availability and cost of these Cyclotrons make them ideal candidates for damaging targets with light ions, rather than neutrons from a neutron source.

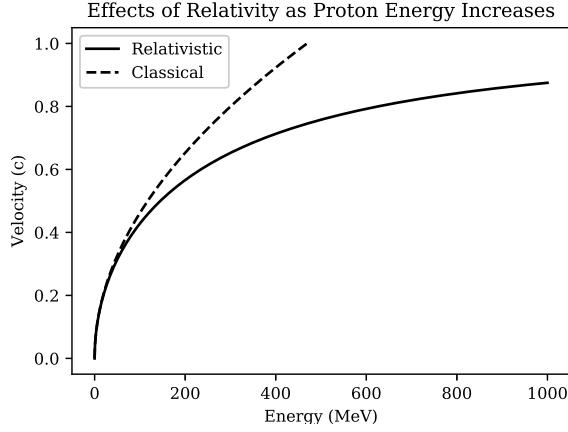


Figure 4.1: A plot of relativistic and classical velocities for 0MeV - 1000MeV protons

The Scanditronix MC40 is an isochronous cyclotron. As the ions are accelerated to higher and higher velocities, the effects of special relativity become appreciable. The relativistic velocity (v) is dependent upon the speed of light (c), the rest energy of the ion (E_0) and the kinetic energy of the particle (E_t). The rest energy is defined by the rest mass (m_0) and the speed of light, although the particle rest masses are typically given in energies of MeV. The difference between the classical and relativistic calculations is illustrated in fig. 4.1.

$$\begin{aligned}
 E_0 &= m_0 c^2 \\
 E_T &= m_0 c^2 \left[\frac{1}{\sqrt{1 - \frac{v^2}{c^2}}} - 1 \right] \\
 \therefore v &= c \sqrt{1 - \left(\frac{E_0}{E_t + E_0} \right)^2}
 \end{aligned} \tag{4.1}$$

The cyclotron has two Dees and these are D shaped hollow electrodes. Ions enter at the center of the cyclotron and are held in the Dees as they are accelerated. The magnetic field is varied to bend the path of the ions from

a circle into a round cornered triangle.

4.1.3 Synchrotron

Two of the most well known accelerators are Synchrotrons: the Large Hadron Collider at Conseil Européen pour la Recherche Nucléaire, and the now retired Tevatron at Fermilab. These are typically large machines and there are approximately 70 around the world[91], approximately a fifth the number of Cyclotrons.

Synchrotrons are used for storing high energy particles in a continuous loop, either to generate light through magnetobremssstrahlung, or for high energy collisions. Cyclotrons are more commonly used for lower energy (tens of MeV) projectiles, in materials science for damaging materials and for medical use (creating radioactive isotopes for use in hospitals).

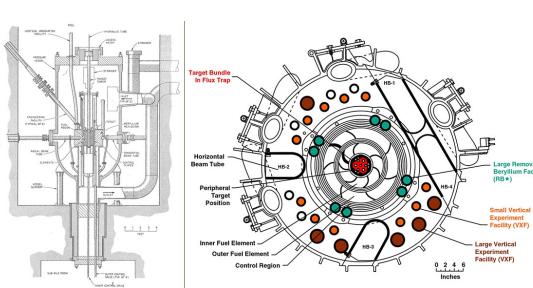
4.2 Neutron Sources

4.2.1 High-Flux Neutron Reactors

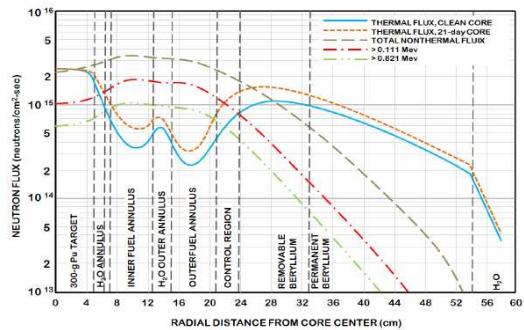
There are 250 or so research reactors in 55 countries [92], and a number of these are used for materials research. In the UK, there is only one remaining research reactor, and this is the Neptune pool type reactor at Rolls Royce[93]. In Cadarache, France, the Jules Horowitz reactor is under construction and this is being built specifically as a materials testing reactor. It will be crucial in researching new materials for use in upcomming Gen IV nuclear power stations [92].

High Flux Isotope Reactor, Oak Ridge

The High Flux Isotope Reactor, at the Oak Ridge National Laboratory in America, is an 85MW research reactor that provides testing space within the reactor as well as a number of neutron beam lines (fig. 4.2). The reactor uses highly enriched Uranium as its fuel source and is scheduled to operate at 100% capacity for 161 days per year, in cycles of approximately 23 days. A 30cm surround of Beryllium is used to reflect neutrons, and in the reactor core there is a high flux of thermal neutrons at a rate of 2.3×10^{15} neutrons $cm^{-2}s^{-1}$ [94].



(a) A cross section of the reactor[95]



(b) Neutron flux in the reactor at 85MW[96]

Figure 4.2: High Flux Isotope Reactor

NIST Center for Neutron Research

The National Bureau of Standards Reactor (fig. 4.3) was designed as a research reactor with a power output of 40MW and a neutron flux of approximately 1.0×10^{15} neutrons $cm^{-2}s^{-1}$. It uses highly enriched Uranium as fuel and is both moderated and cooled by heavy water.

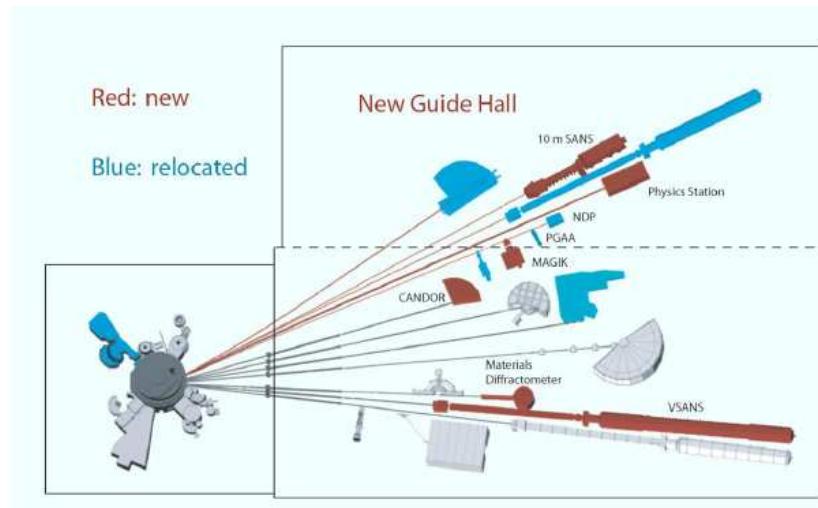


Figure 4.3: NBSR building layout[97]

4.2.2 Spallation

Neutrons result from fission in nuclear reactors, but spallation sources require protons and high mass targets to generate neutrons. The energy of the protons required is a magnitude greater than that of the Scanditronix MC-40 Cyclotron at the University of Birmingham, with spallation source accelerators having a range from 500MeV to over 1GeV.

ISIS is a neutron spallation source located in Harwell, Oxford. Protons are accelerated in a LINAC to 0.37c before being fed into a 163m circumference synchrotron. The protons are then accelerated to 0.84c before being projected, in bunches, at a tungsten target. The impact with the tungsten releases neutrons[98].

Oak Ridge National Laboratory also has a neutron spallation source, the SNS (fig. 4.4). A LINAC accelerates a negatively charged proton (a hydrogen atom with two electrons) up to just below 0.9c. It is stripped of the electrons and enters an accumulation ring as a proton. More protons are accumulated until the beam in the accumulator is diverted at a mercury target. The collision between the mercury target and the protons releases approximately 20 neutrons per collision, and these neutrons pass through beam lines to the experiments .

The source creates a pulse of neutrons from the 50 ton mercury target 60 times per second. The protons are accelerated by the LINAC to between 2.5MeV and 1.0GeV and this results in neutrons with energies almost as high as the proton projectiles. A spectrum of neutron energies is produced with each pulse, although the neutrons may be moderated at the end of the beam lines to slow the neutrons down.

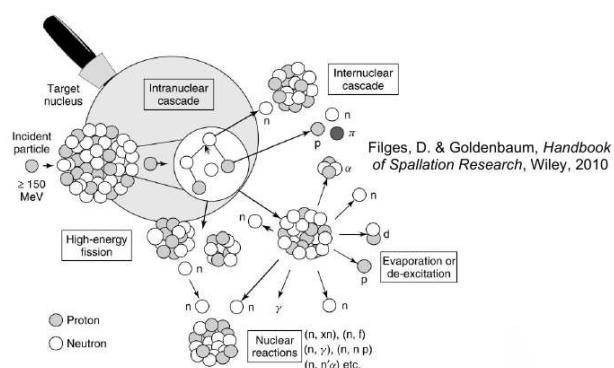


Figure 4.4: An example of a neutron spallation source where high energy ions collide with heavy atoms[99]

4.3 Source Review

Source	Cost	Projectile	Flux/Current	Energy
Scanditronix MC-40 HFIR[96]	£1 million	Proton Neutron	3.7×10^{14} 3.0×10^{15}	8-40MeV Full Spectrum
HFIR[96]		Thermal Neutron	Over 2.0×10^{15}	Thermal
HFIR[96]		Fast Neutron	2.0×10^{15}	$> 0.111\text{MeV}$
HFIR[96]		Fast Neutron	1.0×10^{15}	$> 0.821\text{MeV}$
SNS[100]	\$1.4 billion	Neutron	Average 1.2×10^{13}	Full Spectrum
ISIS spallation source[100]		Neutron	Average 4.0×10^{13}	Full Spectrum

Table 4.2: Examples of neutron sources, energies and flux, with the MC-40 as a reference for protons

The compact proton source that is the cyclotron is compared to neutron sources. It is much smaller and cheaper in comparison and the energy of the projectile may be controlled, whereas a spectrum of energies of neutrons is produced by reactors and spallation sources. Faster neutrons will cause more damage to materials in Gen IV reactors giving the cyclotron an advantage, being able to focus in just one energy range. The three mentioned isotope sources are national facilities that cost much more than a cyclotron. Access is shared between many researchers whereas a cyclotron, such as the Scanditronix MC-40 at the University of Birmingham may have a beam dedicated to a particular task.

4.4 Ion Irradiation to Investigate Neutron Damage

Neutrons emitted during the fission of Uranium-235 have an energy spectra in the intermediate to fast range, with a peak at 1MeV, and a large proportion in the 1MeV to 10MeV range (fig. 4.5). The higher energy neutrons are more of a concern to this work as higher energy neutrons, on colliding with atoms within the target material, cause large damage cascades.

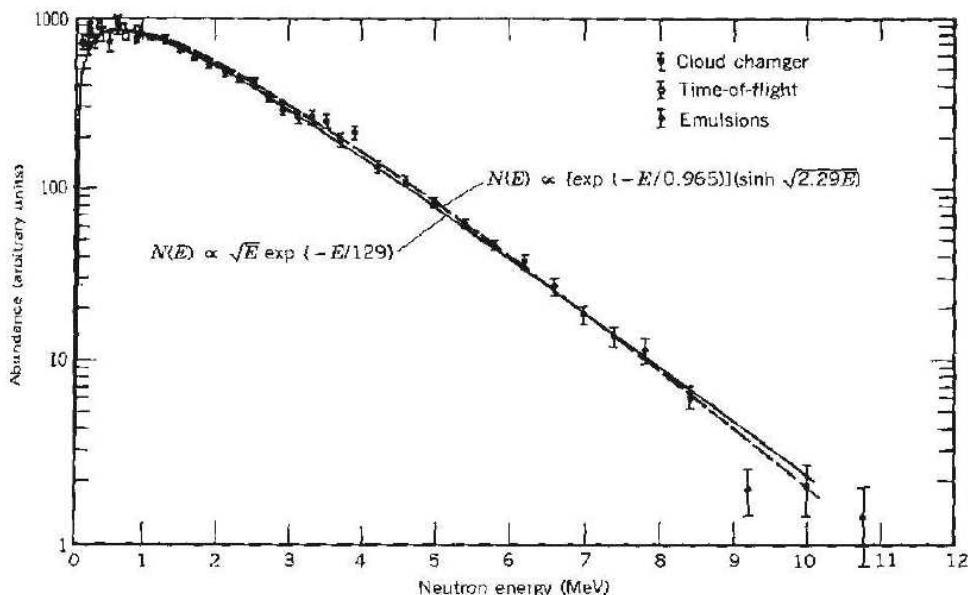


Figure 4.5: Neutron Spectra from the Fission of U235[101]

As discussed earlier in this chapter, there are a number of methods available to create both high energy neutrons and ions, but each has its own set of advantages and disadvantages. At the University of Birmingham high energy ions are created using a cyclotron.

4.4.1 Ion Irradiation at the University of Birmingham

The University of Birmingham acquired a cyclotron in 2004[102], the model being a Scanditronix MC-40. It is capable of accelerating a range of light ions, to energies in the region of 10-60MeV and at currents up to 50 microamps. In the case of protons, this is equivalent to 3.1×10^{14} protons per second[103].

The Cyclotron has been used to provide a steady source of radioactive isotopes for medical applications, including ^{18}F for Positron Emission Particle Tracking (PEPT)[102]. A new beam line was added for materials testing, but for this application the creation of radioactive isotopes is an unwanted by-product of the materials damage research.

The use of neutron sources is typically expensive and the activation of atoms into radioactive isotopes is not impeded by the Coulomb barrier. This is highlighted by cross section data for ^{56}Fe and ^{58}Ni in fig. 4.6. The MC-40 Cyclotron is relatively inexpensive and allows materials research in a more controlled manner, given how the beam may be adjusted in terms of energy and flux as well as being focused onto a smaller area of material.

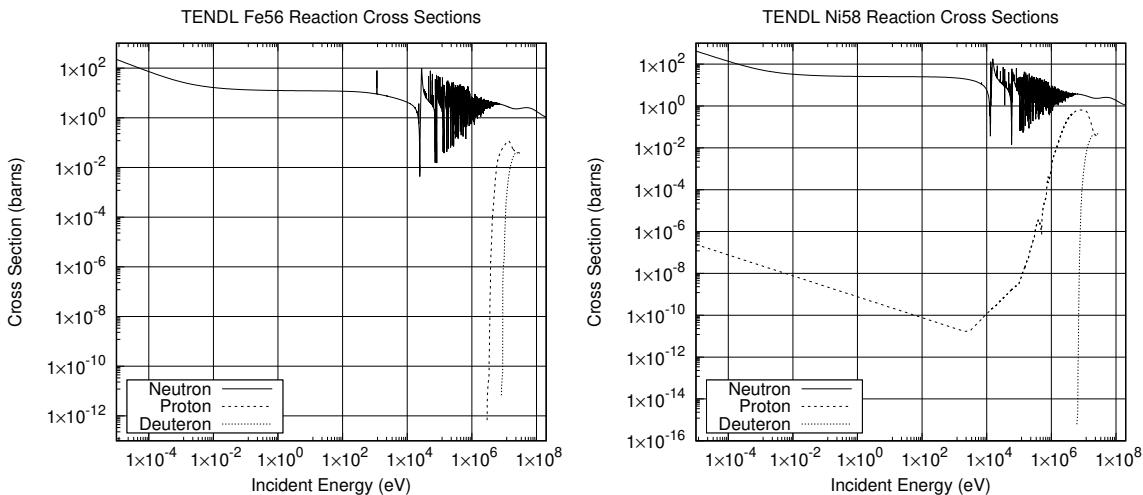


Figure 4.6: Neutron, proton and deuteron cross sections for Fe56 and Ni58, plotted with data from the TENDL database[104]

In chapter 6 the Activity code created in this work will be discussed. It was developed to calculate the activity of targets irradiated by the Cyclotron. It was originally developed in Fortran[103] but was rewritten in Python. Cross section data has been generated using the TALYS code, ion transport is carried out using SRIM and the JEFF [26] nuclear data library is used for radioactive decay data.

4.4.2 Transmutation of Nuclei by Neutrons and Protons

Neutron Activation

The fission of Uranium-235 atoms results in neutrons with a varied spectrum of energies. The neutrons will bounce around inside the reactor losing energy quickly to light atoms within moderators and coolants, such as water. At lower energies the neutrons may be captured by the nuclei of atoms they interact with. This creates a new isotope which may or may not be stable.

Proton Activation

Considering a simplified nuclear potential well, energetic protons approaching a nucleus may overcome the Coulomb potential barrier. They are captured by the nucleus and held within the potential well by the strong nuclear force. The barrier between a proton and a target nucleus may be approximated using eq. 4.2[105] and eq. 4.3[106]. Using these equations does neglect the probability of lower energy protons being captured via quantum tunnelling of the Coulomb barrier. These equations require the charge for the projectile (Z_1) and target (Z_2) and the nuclear radius for each (r_1 and r_2 respectively). In order to approximate the nuclear radius using the Fermi model, the only dependency is the mass number (A).

$$V_{Coulomb} = \frac{e^2}{4\pi\epsilon_0} \frac{Z_1 Z_2}{(r_1 + r_2)} \quad (4.2)$$

$$r = r_0 A^{1/3} \text{ where } r_0 = 1.2 \times 10^{-15} m \quad (4.3)$$

This process may leave the nucleus in an excited and unstable state, depending on the input energy of the proton and configuration of nucleons. The process is probabilistic, and the average chance of a reaction (the microscopic cross section) may be measured as a function of the projectile, projectile energy and target, either experimentally or by optical model potential calculations.

$$R = \frac{J}{Q} \cdot n_p \cdot \sigma \cdot 10^{-28} \delta t \quad (4.4)$$

The reaction rate is calculated from the microscopic cross section, using eq. 4.4, and the parameters are as follows:

- R Reaction Rate (reactions per second)
- J Beam current (A)
- n_p Number density of target (atoms per cubic metre)
- σ microscopic reaction cross section (barns)
- Q projectile charge e.g $1.602177 \times 10^{-19} C$ for a proton
- δt target thickness (m)

The barn (unit) is the standard cross section unit for nuclear physics. As the number density and target thickness (or track length) are given in meters, barns are converted to square meters using the factor 10^{-28} .

4.4.3 Nuclear Reaction Cross Sections

Reaction Cross Sections

The type of reaction for an individual event cannot be determined, but the probability of that reaction happening may be measured and future events predicted. This data may be gathered experimentally, or it may be calculated using various models.

EXchange FORmat Data File

EXchange FORmat (EXFOR) is a standard format for storing experimental results from nuclear reactions[107]. It is also associated with an online database hosted by the International Atomic Energy Agency (IAEA) where nuclear reaction data from a variety of sources are obtained and compared. Thousands of points make up the database and the work of three groups responsible for subsets of that data was reviewed to give an insight into the methods used.

The (p,n) reactions with pure Titanium, Vanadium, Chromium, Iron and Nickel were studied by Tanaka and Furukawa[108]. The 63 inch cyclotron that was used by the Institute for Nuclear Study (INS) in Tokyo provided the protons. The device output a mono-energetic 14MeV beam of protons at a stack of target foils. The Ti, Fe and Ni foils were $2 - 5\text{mg/cm}^2$ in thickness, which is roughly 2-10 micrometers (depending on the density of the material). As the ions passed through the stack of foils their energy dropped and the experimenters estimated the energy using the Bethe stopping power formula[109]. The ion energies in the resulting data range from 3-4MeV up to 14MeV.

Similarly the (p,n) reaction in a range of materials with 5 to 10.5MeV protons was investigated by Wing and Huizenga[110]. A 60 inch cyclotron was used in this instance with foils of Vanadium, Chromium, Copper, Silver, Cadmium and Lanthanum. Foils were again stacked and had a thickness of 5mg/cm^2 , or approximately 5 micrometers, depending on the density of the element. Some energies as they passed through the foils were estimated by the experimenters, based on past work.

Vanadium, Iron and Copper foils were irradiated with 10 to 45MeV protons using the Milan Azimuthally Varying Field (AVF) cyclotron. In this work Gadioli et al[111] measured reaction cross sections for 17 proton energies. Thin foils ranging from 10-48 micrometers but is it unknown whether or not they were stacked or irradiated individually using aluminium absorbers.

The experimental work as a whole ranges over decades using slightly different tools and measuring methods, but the resulting data has contributed to ever improving models used to compute the probability of a reaction.

TALYS and the Optical Model Potential

In the standard model, protons and neutrons are composed of quarks, held together by the strong nuclear force. The nucleus of an atom is also held together by the strong nuclear force that on such small separations overwhelms the electromagnetic force of the protons with one another.

This is a complicated system, not to mention the excited states that nuclei may occupy. The interaction of a projectile (proton, neutron or a heavier ion) with the nucleus is a challenge to model.

Initially, reaction data for protons and neutrons were gathered through experimentation. In the 1950s, Feshbach et al put forward a simple model for nuclear reactions between the nucleus and neutrons, and this model was restricted to 0-20MeV neutrons. The form of the potential used was a complex function[112].

$$\begin{aligned} V &= V_0(1 + i\zeta)r < R \\ V &= 0r > R \end{aligned} \tag{4.5}$$

The potential in equation 4.5 has the parameters $R = 1.45 \times A^{\frac{1}{3}}$, $V_0 = 42\text{MeV}$ and $\zeta = 0.03$ where A is the atomic mass of the target atom. These parameters were fit to the experimental data available to Feshbach et al[113] with the only variable being the atomic mass[112].

Considering the complexity of the system being modelled, this simple model was very successful. Over the years since, the models used have become more complex and parameters used have been fit to an increasing amount of experimental data.

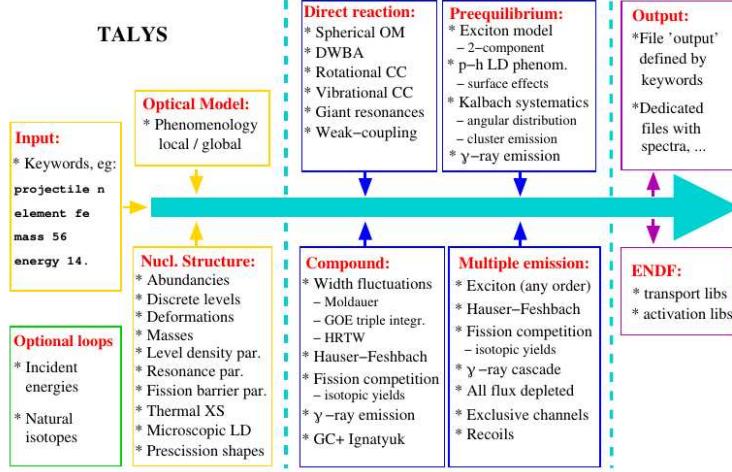


Figure 4.7: TALYS work flow [114]

The TALYS code uses a range of models (fig. 4.7). These include the optical model which is solved using the Equations Couplées Itérations Séquentielle (ECIS-06) code of Jacques Raynal, implemented as a subroutine within TALYS, and this is accurate up to 180MeV[114]. Whilst the TALYS code has been extended up to 1GeV, it is experimental in the 180MeV to 1GeV range. Fortunately, this work only requires nuclear reaction cross section data up to approximately 100MeV as the current ion source under consideration produced ions with energies up to 60MeV.

The TALYS code has potentials for protons and neutrons, but it also has potentials for deuterons, tritons, helium-3 and alpha particles. The potentials are discussed in detail in the TALYS manual[114]. This extension to heavier ions may be useful for this work as the University of Birmingham cyclotron is capable of accelerating deuterons and He^{2+} ions.

Comparing experimental data to the TALYS data for $Fe54(p,\gamma)Co55$ shows good agreement between 1MeV and 10MeV. There was insufficient experimental data from this source, but the TALYS generated data ranges from micro eV up to 100MeV+, as seen in fig. 4.8[115].

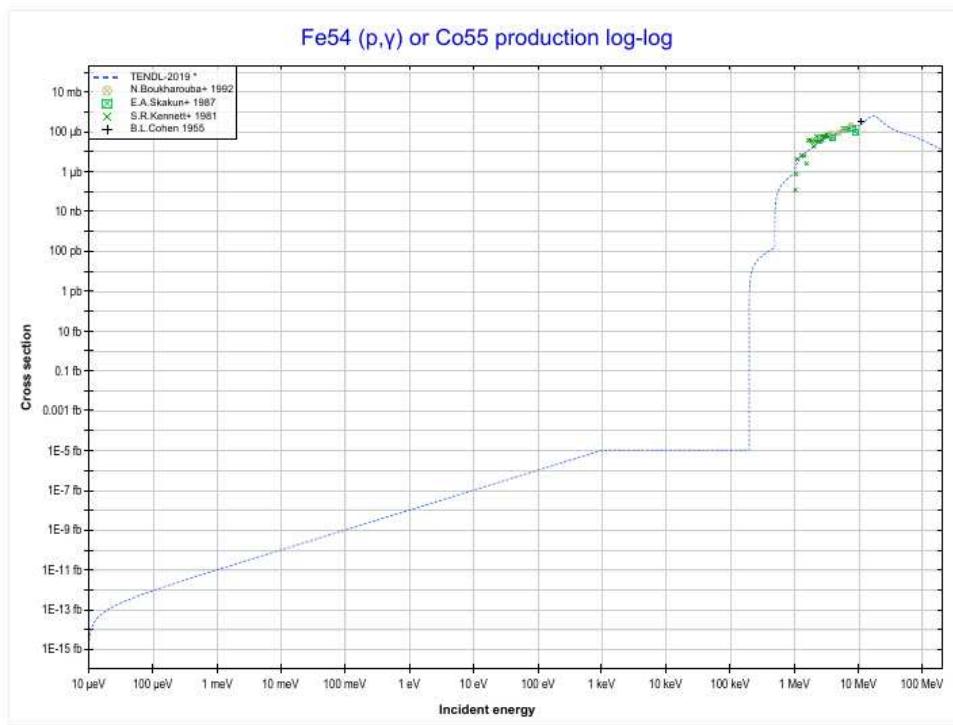


Figure 4.8: TALYS Fe54-Co55 cross section comparison with experiment [115]

Proton Activation Data File

The Proton Activation Data File (PADF) was released in 2007 and contained nuclear reaction data for 2355 target nuclei, ranging from Magnesium (12) to Radon (86) with proton energies up to 150MeV. The file contains the sum of all individual reactions as well as certain yields, and the data was generated using the TALYS and ALICE/ASH codes, as well as experimental data from Exfor[116].

Listing 4.1: A sample of the Iron-56 PADF data file

```

1      Proton Activation Data File          528 1451 12
2      ****
3      *****                                     528 1451 13
4      *****                                     528 1451 14
5      Authors: C.H.M.Broeders, U.Fischer, A.Yu.Konobeyev, L.Mercatali 528 1451 15
6      *****                                     528 1451 16
7      *****                                     528 1451 17
8      Data for PADF file were obtained using the TALYS code [1] for 528 1451 18
9      target nuclei with half-life more than 10 min and using the 528 1451 19
10     ALICE/ASH code [2] for nuclei with 1 sec < T1/2 < 10 min and 528 1451 20
11     available experimental data           528 1451 21
12     MAT numbers are taken according to JEFF-3.1/RDD       528 1451 22
13     *****                                     528 1451 23
14     Evaluation for Fe-56 (stable) : TALYS code          528 1451 24
15     Experimental data used for the correction of calculated 528 1451 25
16     excitation functions are taken from Refs.[3-12]        528 1451 26
17
18     File contains                           528 1451 27
19
20     MF=3 MT=5                            528 1451 28
21     Sum of all individual reaction cross-sections (Total 528 1451 29
22     reaction cross-section)                 528 1451 30
23
24     MF=6 MT=5                            528 1451 31
25     Yields of nuclei in nuclear reactions including 528 1451 32
                                         528 1451 33
                                         528 1451 34
                                         528 1451 35
                                         528 1451 36

```

TENDL 2019 Data Files

The cross section data for protons and neutrons is available to download in ENDF format files. The data sources used by this work are the TALYS code and TENDL data files, and these are created by a combination of different nuclear models and experimental data. The TENDL nuclear reaction simulation code provides the calculated data.

The nuclear reaction files are rather large, and they all follow a standard format.

Listing 4.2: Sample TENDL File

1	2.605600+4	5.545443+1	0	0	0	02631	3	2	1
2	0.000000+0	0.000000+0	0	0	1	462631	3	2	2
3	46	2				2631	3	2	3
4	1.000000+3	-9.920042-7	1.000000+6	-9.920042-7	2.000000+6	-2.167241-42631	3	2	4
5	3.000000+6	-7.609736-3	4.000000+6	-3.232345-2	5.000000+6	-5.975554-22631	3	2	5
6	6.000000+6	-5.149510-2	7.000000+6	-3.685484-2	8.000000+6	-5.542173-22631	3	2	6
7	9.000000+6	-9.978765-2	1.000000+7	-1.599116-1	1.100000+7	-2.206871-12631	3	2	7
8	1.200000+7	-2.743084-1	1.300000+7	-3.165007-1	1.400000+7	-3.471231-12631	3	2	8
9	1.500000+7	-3.674350-1	1.600000+7	-3.788282-1	1.700000+7	-3.825634-12631	3	2	9
10	1.800000+7	-3.799978-1	1.900000+7	-3.725944-1	2.000000+7	-3.617662-12631	3	2	10
11	2.200000+7	-3.342054-1	2.400000+7	-3.031296-1	2.600000+7	-2.709398-12631	3	2	11
12	2.800000+7	-2.388019-1	3.000000+7	-2.077513-1	3.500000+7	-1.398614-12631	3	2	12
13	4.000000+7	-8.801994-2	4.500000+7	-5.013441-2	5.000000+7	-2.349373-22631	3	2	13
14	5.500000+7	-5.480233-3	6.000000+7	-6.189980-3	6.500000+7	-1.332313-22631	3	2	14
15	7.000000+7	-1.727774-2	7.500000+7	-1.906833-2	8.000000+7	-1.943711-22631	3	2	15
16	9.000000+7	-1.783081-2	1.000000+8	-1.499871-2	1.100000+8	-1.211197-22631	3	2	16
17	1.200000+8	-9.641120-3	1.300000+8	-7.717424-3	1.400000+8	-6.318436-32631	3	2	17
18	1.500000+8	-5.367555-3	1.600000+8	-4.771939-3	1.800000+8	-4.312985-32631	3	2	18
19	2.000000+8	-4.445323-3				2631	3	2	19

4.4.4 Radioactive Decay

Radioactive decay is the random change in nucleons or energy state of an unstable nucleus. It is impossible to predict when a single nucleus will decay, but the decay of a collection of nuclei is statistical in nature. The radioactivity and number of unstable nuclei at time (t) can be predicted using the decay constant (λ) for the radioactive isotope. This constant is defined as follows:

$$\lambda = -\frac{N'(t)}{N(t)} \quad (4.6)$$

The number of radioactive nuclei $N(t)$ at time t is given by the following equation, where $N(0)$ is the starting number of nuclei:

$$N(t) = N(0) \exp(-t\lambda) \quad (4.7)$$

The activity $A(t)$ of the radioactive nuclei is predicted at time t by using the following equations, where $N'(t)$ is the change in amount of nuclei with respect to time:

$$A(t) = -N'(t) = \lambda N(t) \quad (4.8)$$

$$A(t) = \lambda N(0) \exp(-t\lambda) \quad (4.9)$$

4.4.5 Saturation Activity

As a radioactive isotope is created by reactions between target atoms and projectiles (protons, neutrons, deuterons) it will begin to decay. The amount of the isotope will continue to increase until the decay rate equals the reaction rate for the creation of the isotope.

For a single isotope the production rate (ω) is calculated using the ion current (J), the charge per ion (Q), the number density of atoms in the target (n_p), the reaction cross section (σ) and the track length (δt), as given in eq. 4.10. The units are amps and coulombs for the current and charge, whereas the cross section is in barns and the number density in atoms per cubic meter. The track distance is measured in meters. This necessitates the conversion from barns to square meters, hence the factor 10^{-28} .

$$\omega = \frac{J}{Q} \cdot n_p \cdot \sigma \cdot 10^{-28} \delta t \quad (4.10)$$

The change in the amount of an isotope with time ($\frac{dN}{dt}$) is the source rate minus the loss rate (λN) as given in eq. 4.11. This is given as the initial condition the starting amount N_0 at time $t = 0$.

$$\frac{dN}{dt} = \omega - \lambda N, N(0) = N_0 \quad (4.11)$$

This equation may be solved using Laplace transforms (appendix A) and this will be explored in more detail in section 6.2.1. The solution for a single isotope that has a starting amount N_0 , a decay constant λ and a creation rate of ω is given in eq. 4.12.

$$N(t) = \frac{\omega}{\lambda} + \left(N_0 - \frac{\omega}{\lambda} \right) \exp(-\lambda t) \quad (4.12)$$

The saturation activity may be computed as follows. As t approaches infinity the exponential term tends towards zero due to the negative decay constant. Replacing the exponential terms with zero and noting eq. 4.8 leaves a simple result to compute the saturation activity (A_s) (eq. 4.14). This is logical as, once saturated, the isotope will decay at the same rate as its source.

$$N_s = \frac{\omega}{\lambda} \\ A_s = \lambda N_s = \omega \quad (4.13)$$

The saturation time lies on an asymptote, therefore the saturation time is undefined. Rather than attempt to compute this time it is more useful to compute the time at which the saturation reaches a fraction (k) of the maximum saturation amount (N_s).

$$kN_s = \frac{\omega}{\lambda} + \left(N_0 - \frac{\omega}{\lambda} \right) \exp(-\lambda t) \\ t = \frac{-1}{\lambda} \ln \left[\frac{k-1}{N_0 \frac{\lambda}{\omega} - 1} \right] \quad (4.14)$$

The equation is useful in calculating (near to) the maximum possible activity of a isotope being created in a proton beam or other similar process where radioactive isotopes are created and lost at a constant rate.

4.4.6 Decay Constants: Joint Evaluated Fission and Fusion File (JEFF) 3.3 Data File

JEFF 3.3 is an evaluated data file[26], meaning it has been evaluated by a relevant authority. The quality of the data may affect the health of the public and industry workers, so it must be evaluated. This particular file is managed by and available through the Nuclear Energy Agency (NEA).

It is a collection of many data files. Released in 2017, it also contains several files for incident gammas, protons, deuterons, tritons, helium-3 and alphas from the TENDL 2017 data file.

The files that will be important for this work are the ENDF-6 radioactive decay data files only. The nuclear reaction cross section data will come from TALYS and various iterations of the TENDL libraries. The decay data held in the JEFF 3.3 file includes isotope data, masses, half lives, branching factors, continuous and discrete gamma data.

4.4.7 Bateman Equation for Radioactive Decay

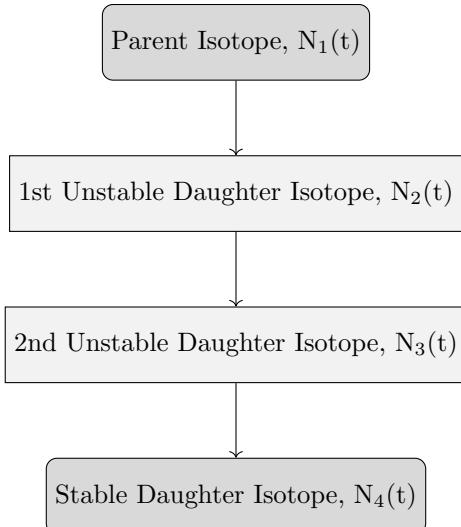


Figure 4.9: An example decay chain from an unstable parent isotope, through unstable daughter isotopes ending with a stable daughter isotope.

The English mathematician Harry Bateman derived an equation (eq. 4.15[1][117]) to calculate the amount of each isotope in a decay chain, illustrated in fig. 4.9, at time t. N is the isotope amount and n denotes the nth isotope in that decay chain. The decay constant (λ) is required for each isotope in the decay chain along with the starting amount for the first isotope in the decay chain ($N_1(0)$).

$$N_n(t) = \frac{N_1(0)}{\lambda_n} \sum_{i=1}^n \left[\prod_{j=1, j \neq i}^n \frac{\lambda_j}{\lambda_j - \lambda_i} \right] \exp(-\lambda_i t) \quad (4.15)$$

This equation (eq. 4.15) allows the user to calculate how much of each isotope there is expected to be at some time in the future, where each of the isotopes in the decay chain is decaying.

When a radioactive isotope decays, there may be more than one mode of decay, and this leads to branching factors. Pb-214 only decays via beta decay to Bi-214, giving a branching factor of 1.0, whereas Bi-214 has a

99.979% chance of decaying to Po-214 by beta decay and a 0.021% of emitting an alpha particle and decaying to Tl-210 (branching factors of 0.99979 and 0.00021 respectively) [26].

When a target material is irradiated, there is a source term for transmuted nuclei due to the irradiation. The daughter isotopes of these transmuted isotopes will also be affected by the irradiation and will transmute further, giving a source term for each daughter isotope as a result of the irradiation. Sources for each isotope in the decay chain, and branching factors between a parent isotope and its daughter isotope/s must be accounted for. This is addressed in chapter 6.

4.5 Simulating Ion Irradiation with SRIM and TRIM

A package of ion transport codes, SRIM, is freely available to download and use to investigate the transport of ions through matter. It includes tools to calculate the stopping range of a given ion in a material as well as TRansport In Matter, a code used to calculate the energy and position of an ion through a target.

TRIM does not take into account the structure of a target. It may be layered in the calculation perpendicular to the beam, but beyond that the target is treated as amorphous. It will give the same result for BCC iron as it would for FCC iron, providing the density remains the same. The density and whether or not the target layer is a gas or solid are options that must be configured when setting up a calculation. Each ion history is tracked through the same target; the target never changes. TRIM is ignorant to the structure and therefore damage to the structure that would accumulate over time, and to any swelling and change in composition or density.

The interaction of the ion with the target material is split into two parts: nuclear loss and electronic loss. The ion is tracked through the target until either its energy drops below a set threshold (10eV) or it leaves the target (fig 4.10). The code then moves on to the next ion history.

In a projectile-nucleon interaction a number of equations were developed by Biersack et al[118] where a number of parameters are fit for the screened potential. They are used to determine the scattering angle Θ of the ion and target nucleus, as well as the amount of energy transferred. The azimuthal angle is selected randomly by multiplying 2π by a random float [0.0, 1.0].

The screening length (a) is dependent upon the charge of the projectile (Z_1) and that of the target (Z_2) (eq. 4.16).

$$a = \frac{0.8853a_0}{(Z_1^{\frac{2}{3}} + Z_2^{\frac{2}{3}})} \quad (4.16)$$

$$\epsilon \equiv \frac{aE_c}{Z_1 Z_2 e^2} \quad (4.17)$$

The scattering angle (Θ) is computed using eq. 4.18. It is dependent upon the screening length (eq. 4.16) as well as the reduced energy (ϵ) (eq. 4.17). Five other parameters were fit (C_1 to C_5) by Biersack and these are given in eq. 4.19.

$$\cos\left(\frac{\Theta}{2}\right) = \frac{B + R_c + \Delta}{R_0 + R_C}$$

where

$$B = \frac{p}{a}, R_0 = \frac{r_0}{a}, R_C = \frac{\rho}{a}$$

$$\Delta = A \frac{R_0 - B}{1 + G}$$
(4.18)

$$A = 2\alpha\epsilon B^\beta$$

$$G = \gamma \left((1 + A^2)^{\frac{1}{2}} - A \right)^{-1}$$

$$\alpha = 1 + C_1\epsilon^{-\frac{1}{2}}, \beta = \frac{C_2 + \epsilon^{\frac{1}{2}}}{C_3 + \epsilon^{\frac{1}{2}}}, \gamma = \frac{C_4 + \epsilon^{\frac{1}{2}}}{C_5 + \epsilon^{\frac{1}{2}}}$$

where p is the impact parameter

where a is the screening length

where r_0 is the distance of closest approach

where ϵ is the reduced energy

(4.19)

The potential between the projectile and target is calculated at the distance of closest approach, r_0 . It is a Coulomb type potential (eq. 4.20) than includes the universal screening function (eq. 4.21)[118]. This potential will appear again, although occasionally with differing parameters, in chapter 5. The parameters defined here in eq. 4.21 are those derived and published by Ziegler, Ziegler and Biersack in the 2015 SRIM book[118].

$$V(R) = \frac{Z_1 Z_2 e^2}{a R} \Phi(R)$$
(4.20)

$$\Phi(R) = 0.1818 \exp(-3.2x) + 0.5099 \exp(-0.9423x) + 0.2802 \exp(-0.4028x) + 0.2817 \exp(-0.2016x)$$
(4.21)

The recoil nuclei are also followed through several generations until their energies fall below that set for either the surface binding energy or displacement energy (3-6eV and 15-30eV respectively)[118].

The free flight path between large collisions is calculated based on the projectile energy, type and the target material. The path length between collisions is randomly generated by taking the maximum free path and multiplying by a random float [0.0, 1.0]. Smaller projectile-nucleus interactions are not individually calculated, but the average energy that would have been lost is calculated and applied.

The ion continuously loses energy to the electrons in the material. TRIM calculates this over the free flight path, L (eq. 4.22).

$$\Delta E_e = L N S_e(E)$$

where $S_e(E)$ is the electronic stopping

and N is the atomic density of the target

(4.22)

The target damage is a combination of the incoming ion, the primary knock on atoms and their damage cascades. The user has the option of running a full damage cascade that plots the entire cascade, or a quicker calculation

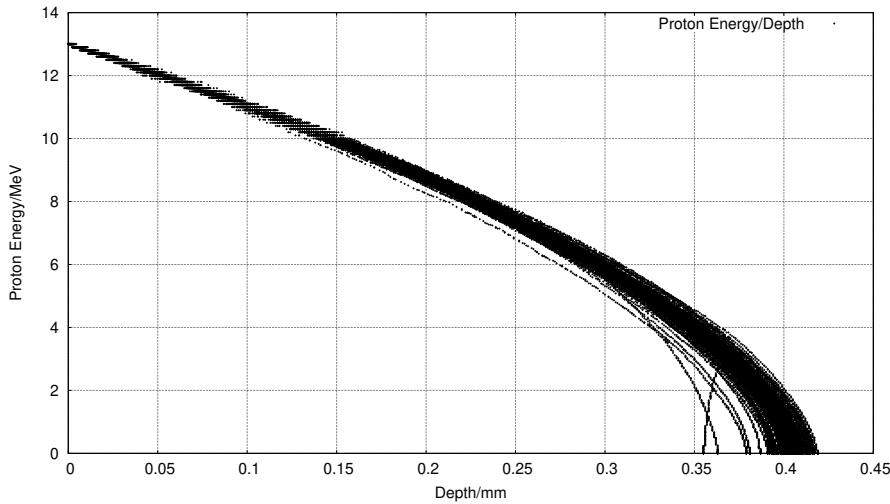


Figure 4.10: One hundred simulated 13MeV proton energy loss curves in Fe simulated with SRIM [32]

where the cascade is treated as just a point. The total number of displacements is the sum of vacancies and replacement collisions, and the vacancies is sum of interstitials with the number of atoms that leave the target.

One file that SRIM creates is of importance to the Activity code, and that is the trajectory file that contains the energy and x,y,z co-ordinate data points for simulated ions moving through matter. Fig. 4.10 shows the trajectory of one hundred 13MeV protons entering and passing through an Iron target, and it is this set of data points (together with the cross section database) that the Activity code uses to calculate the reaction rates for the transmutation of nuclei in the target. At higher energies, the ions slow as they lose energy due to electronic stopping, but as the ion energy drops the mechanism of loss through nuclear collisions becomes important. The spreading of ion depths at lower energies is a result of the higher momentum transfer during nuclear collisions.

Chapter 5

Interatomic Potential Fitting

In order to model Fe, Pd and Ru with Molecular Dynamics codes, an interatomic potential is needed. This will require experimental data and data generated by first principles calculations. The simplified models will consist of only Fe-Pd and Fe-Ru, but pure Fe does not take the FCC structure found when alloyed with Ni in Austenitic Stainless Steel.

The properties of theoretical pure FCC Fe are estimated using Density Functional Theory that solves the many body Schrödinger Equation. While it is impossible to solve this equation exactly, with our current technology and knowledge, it is possible to solve approximately using the exact theory of Hohenberg-Kohn, the Kohn-Sham equations and a number of simplifications.

Many body potentials such as the Finnis-Sinclair (FS) and embedded-atom method (EAM) are often used to model metals. The force matching method is used to fit the potentials to data. Optimization algorithms are split into global and local, with an example of a global being simulated annealing and a local being BFGS.

5.1 Experiment, Modelling and Theory

5.1.1 Introduction

Quantum Mechanics is a successful theory but solving the many electron Schrödinger equation for more than a couple of electrons soon becomes impossible with current technology. Carrying out experiments may be the most genuine way to investigate a material, but damage processes occur over an incredibly wide range of time scales. Using computers to model a system helps to bridge the gap between theory and experiment and helps to give insights into materials.

5.1.2 Simulating Materials on a Variety of Scales in Time and Space

A single one micron grain of steel would contain tens of billions of atoms. This is too large a number of atoms to model currently, but by choosing the right type of model a portion of the grain may be modelled.

The properties of metals may be approximated using small crystals rather than the whole grain, and this is ideally suited to DFT calculations. Simulations of small ensembles of atoms over time are now also possible using DFT molecular dynamics. Larger simulations containing thousands to millions of atoms, that are able to simulate single damage cascades are suitable for MD simulations (fig. 5.1).

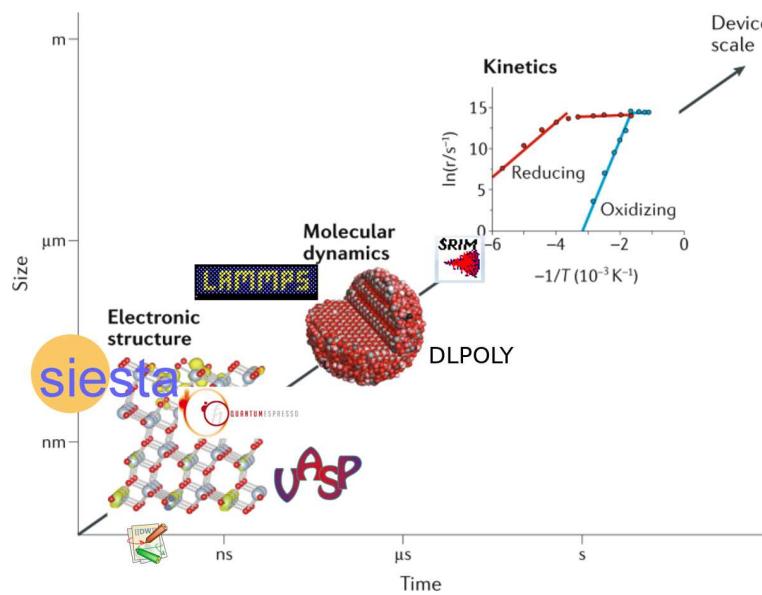


Figure 5.1: Time and Size Scales for Computer Packages [119]

5.2 Properties of Crystals

5.2.1 Introduction

There are seven crystal classes (table. 5.1) and 14 Bravais lattices (section 5.7.6), although this work is primarily concerned with cubic and orthorhombic crystals.

Class	Lengths	Angles
Cubic	$a = b = c$	$\alpha = \beta = \gamma = 90^\circ$
Hexagonal	$a = b, c$	$\alpha = \beta, \gamma = 120^\circ$
Rhombohedral	$a = b = c$	$\alpha = \beta = \gamma \neq 90^\circ$
Tetragonal	$a = b, c$	$\alpha = \beta = \gamma = 90^\circ$
Orthorhombic	a, b, c	$\alpha = \beta = \gamma = 90^\circ$
Monoclinic	a, b, c	$\alpha = \beta = 90^\circ, \gamma \neq 90^\circ$
Triclinic	a, b, c	$\alpha, \beta, \gamma,$

Table 5.1: Seven Crystal Classes

Metals are rarely single crystals, but are made from many crystals separated by grain boundaries. However, the knowledge of the microscopic crystals translates well to the properties of the metals on a macroscopic scale.

This work does not attempt to calculate properties based on a collection of many crystals, but on a single crystal with less than a thousand atoms. The crystals are then assumed to be infinite in size, due to periodic boundary conditions.

5.2.2 Equation of State

The equation of state relates two or more intensive thermodynamic parameters. In this work, the parameters of interest are the energy and volume of a system and the equation of state that connects them. Knowing this not only allows one to predict the energy or pressure at a certain volume, but also the minimum energy, relaxed volume and the bulk modulus. Being able to determine these equations for materials will be important when

fitting interatomic potentials as they will be used to contribute to measuring how well the model, using the derived potentials, matches either DFT or experimental values.

Bulk Modulus

The bulk modulus of a material is defined as the bulk stress of a sample divided by the bulk strain on that sample and several example values are given in table 5.2. It is also the inverse of the compressibility of that material, which means that materials with a higher bulk modulus are less compressible than those with a lower value.

$$B_0 = -V \frac{\partial P}{\partial V} \quad (5.1)$$

$$B_0 = V \frac{\partial^2 E}{\partial V^2} \quad (5.2)$$

The bulk modulus (B_0) may be calculated using the change in pressure (P) with respect to volume (V), as given in eq. 5.1. It may also be computed using the second derivative of the energy (E) with respect to volume (eq. 5.2).

Material	Bulk Modulus/GPa
Aluminium	70
Iron (BCC)	110
Stainless Steel 18-8	163

Table 5.2: Example bulk modulii for three metals

Murnaghan Equation of State

Hooke's law implies a linear relationship between stress and strain. In practice, where a pressure is applied to a material, the application of Hooke's law is limited. Muraghan derived a new equation[120] to improve upon formulae developed in the 1930's, using compression data from high pressure experiments.

$$P(V) = \frac{B_0}{B'_0} \left(\left(\frac{V_0}{V} \right)^{B'_0} - 1 \right) \quad (5.3)$$

As pressure is the negative derivative of the internal energy of the system with respect to change in volume, $P = -(\partial E / dV)$ (eq. 5.3, the equation can be integrated and written in terms of the energy, volume, bulk modulus and its derivative (B'_0))[121].

$$E(V) = E_0 + \frac{B_0 V}{B'_0} \left[\left(\frac{V_0}{V} \right)^{B'_0} \frac{1}{B'_0 - 1} + 1 \right] - \frac{B_0 V_0}{B'_0 - 1} \quad (5.4)$$

The relaxed energy (E_0) and relaxed volume (V_0) are also required and the result is given in eq. 5.4.

Birch-Murnaghan Equation of State

Several years after Murnaghan's equation, Birch developed the equation of state (eq. 5.5) further upon the experimental data provided by work from Bridgman in high pressure physics. For cubic symmetry, the description of free energy now includes third order terms in the strain components[122].

$$P(V) = \frac{3B_0}{2} \left[\left(\frac{V_0}{V} \right)^{\frac{7}{3}} - \left(\frac{V_0}{V} \right)^{\frac{5}{3}} \right] \left[1 + \frac{3}{4}(B'_0 - 4) \left(\left(\frac{V_0}{V} \right)^{\frac{2}{3}} - 1 \right) \right] \quad (5.5)$$

The energy-volume relationship may again be constructed[121] (eq. 5.6).

$$E(V) = E_0 + \frac{9V_0B_0}{16} \left[\left[\left(\frac{V_0}{V} \right)^{\frac{2}{3}} - 1 \right]^3 B'_0 + \left[\left(\frac{V_0}{V} \right)^{\frac{2}{3}} - 1 \right]^2 \left[6 - 4 \left(\frac{V_0}{V} \right)^{\frac{2}{3}} \right] \right] \quad (5.6)$$

In order to fit the Birch-Murnaghan, the first step is to fit a second order polynomial to the energy-volume data. This may be achieved using least-squares fitting with a Vandermonde matrix. The coefficients from this polynomial may then be used to calculate reasonable values for E_0 , V_0 and B_0 ; same starting values of B'_0 are between 1 and 10, and a recommended value in the literature is 2[123].

$$\begin{aligned} E(V) &= c_0 + c_1 V + c_2 V^2 \\ V_0 &= -\frac{c_1}{2c_2} \\ E_0 &= c_2 \times V_0^2 + c_1 V_0 + c_0 \\ B_0 &= 2c_2 V_0 \\ B'_0 &= 2 \end{aligned} \quad (5.7)$$

A suitable optimization algorithm (Newton-Raphson, Newton-Gauss, BFGS) may then be used to attempt to minimise the fit further for E_0 , V_0 and B_0 whilst trying values of $B'_0 \in 1.0, 1.5, 2.0, 2.5, 3.0, 3.5, 4.0, 4.5, 5.0, 5.5, 6.0$. The parameters with the lowest residual square sum are returned.

Mehl et al used a version of the Birch equation of state[122] and suggest a value for B'_0 between 3 and 5[124] and limit the value of N in eq. 5.8 to either 3 or 4 for the best data fit.

$$E(V) = E_0 + \frac{9}{8}B_0V_0 \left[\left(\frac{V_0}{V} \right)^{\frac{2}{3}} - 1 \right]^2 + \frac{9}{16}B_0(B'_0 - 4)V_0 \times \left[\left(\frac{V_0}{V} \right)^{\frac{2}{3}} - 1 \right]^3 + \sum_{n=4}^N \gamma_n \left[\left(\frac{V_0}{V} \right)^{\frac{2}{3}} - 1 \right]^n \quad (5.8)$$

In generating the data for a Birch or Birch-Murnaghan fit the atom configuration is relaxed. A small strain $[\sigma, \sigma, \sigma, 0.0, 0.0, 0.0]$ is applied to decrease and increase the volume slightly about the relaxed volume.

Rose-Vinet Equation of State

More recently a Universal form, eq 5.9, was proposed by Rose, Vinet et al[125][126]. With an exponential cut off, this fits from the relaxed lattice parameter to much wider separations with a good fit for simple and transition metals, including K, Mo, Cu (fig. 5.2).

$$E(a) = E_0 + \left(1 + -\alpha \left(\frac{a}{a_0 - 1}\right)\right) \exp\left(-\alpha \left(\frac{a}{a_0 - 1}\right)\right) \quad (5.9)$$

$$\alpha = \sqrt{\frac{9\Omega B_0}{-E_0}}$$

The energy ($E(a)$) is dependent upon the lattice parameter (a). The relaxed lattice parameter (a_0), cohesive energy (E_0), atomic volume of the relaxed primitive cell (Ω) and bulk modulus (B_0) are required for this equation of state.

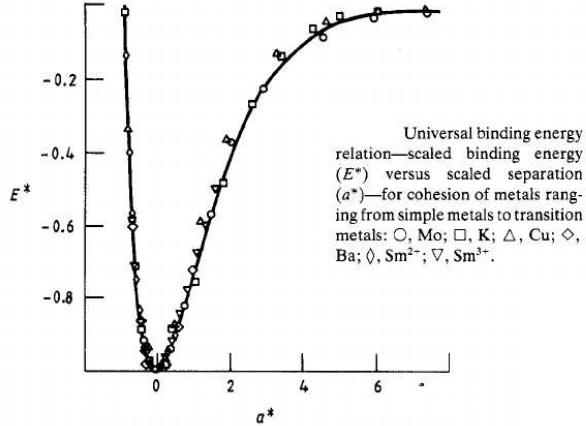


Figure 5.2: Rose-Vinet Equation of State[125]

This has been used as a tool to fit interatomic potentials in work by Jelinek et al[127] and Bonny et al[128]. The version of the equation in this work is that used by Bonny et al and their fitting code, which will be discussed in more detail in section 5.6.

5.2.3 Voigt Notation

Where a tensor is symmetric, Voigt notation is used to simplify how the tensor is written. It reduces a second order tensor such as stress or strain, represented by a 3x3 matrix, to a 6 row matrix. It also reduces a fourth order tensor, such as the compliance or stiffness tensor (represented by a 3x3x3x3 matrix), to a 6x6 matrix.

$$\vec{A} = \begin{bmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{bmatrix} = \begin{bmatrix} A_{11} \\ A_{22} \\ A_{33} \\ A_{23} \\ A_{13} \\ A_{12} \end{bmatrix} \text{ if } \vec{A} \text{ is symmetric} \quad (5.10)$$

5.2.4 Elastic Constants

Stress and strain are both related by either the stiffness tensor C , filled with the elastic constants, or the compliance tensor S and this is the inverse of the stiffness tensor. Strain is a measure of the deformation of a material and by definition has no units. In one dimension $\epsilon = \frac{\delta l}{l_0}$ where l_0 is the unstrained length and δl is the

change in length. The Generalized Hooke's law relating these tensors allows the computation of the Cauchy stress if the elastic constants are known and an input strain is provided.

$$\vec{A} = \begin{bmatrix} \epsilon_{11} & \epsilon_{12} & \epsilon_{13} \\ \epsilon_{21} & \epsilon_{22} & \epsilon_{23} \\ \epsilon_{31} & \epsilon_{32} & \epsilon_{33} \end{bmatrix} = \begin{bmatrix} \epsilon_{11} \\ \epsilon_{22} \\ \epsilon_{33} \\ \epsilon_{23} \\ \epsilon_{13} \\ \epsilon_{12} \end{bmatrix} \text{ where } \epsilon_{ij} \equiv \frac{1}{2} \left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) \quad (5.11)$$

$$\vec{\sigma}_{ij} = \vec{C}_{ijkl} \vec{\epsilon}_{kl} \quad (5.12)$$

$$\vec{\epsilon}_{ij} = \vec{S}_{ijkl} \vec{\sigma}_{kl} \quad (5.13)$$

$$\epsilon_{ij} = \begin{bmatrix} \epsilon_1 = \epsilon_{11} = \epsilon_{11} \\ \epsilon_2 = \epsilon_{22} = \epsilon_{22} \\ \epsilon_3 = \epsilon_{33} = \epsilon_{33} \\ \epsilon_4 = \epsilon_{23} = \epsilon_{32} \\ \epsilon_5 = \epsilon_{13} = \epsilon_{31} \\ \epsilon_6 = \epsilon_{12} = \epsilon_{21} \end{bmatrix} \quad (5.14)$$

$$\begin{bmatrix} \sigma_1 \\ \sigma_2 \\ \sigma_3 \\ \sigma_4 \\ \sigma_5 \\ \sigma_6 \end{bmatrix} = \begin{bmatrix} C_{11} & C_{12} & C_{13} & C_{14} & C_{15} & C_{16} \\ C_{21} & C_{22} & C_{23} & C_{24} & C_{25} & C_{26} \\ C_{31} & C_{32} & C_{33} & C_{34} & C_{35} & C_{36} \\ C_{41} & C_{42} & C_{43} & C_{44} & C_{45} & C_{45} \\ C_{51} & C_{52} & C_{53} & C_{54} & C_{55} & C_{55} \\ C_{61} & C_{62} & C_{63} & C_{64} & C_{65} & C_{66} \end{bmatrix} \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \epsilon_3 \\ \epsilon_4 \\ \epsilon_5 \\ \epsilon_6 \end{bmatrix} \quad (5.15)$$

There are 81 elastic constants, although as the tensor is symmetric it may be written in Voigt notation with 36 values (eq. 5.15). The number of independent elastic constants may be reduced further depending on the symmetry of the crystal. The tensors in Voigt notation are given for cubic, orthorhombic/orthotropic, monoclinic and triclinic in appendix I.

5.2.5 Calculating Elastic Constants for a Cubic Crystal

Cubic crystals are the most simple class with primitive unit cells having three orthonormal basis vectors. Four of the most common variants of the cubic crystal are the simple cubic, body centered cubic, face centered cubic and zincblende. Due to symmetry, a cubic crystal has only three elastic independent constants; in Voigt notation these are C_{11} , C_{12} and C_{44} .

Applying two strains to a cubic crystal [129] coupled with the calculation of the bulk modulus, as already discussed, allows the three independent elastic constants to be calculated.

First, the bulk modulus may be calculated either by finding the second derivative of the energy wrt volume at the relaxed volume, $B(V) = -VP'(V) = VE''(V)$ [129], or by fitting the Birch-Murnaghan equation of state.

$$\epsilon_{(C11-C22)} = \begin{bmatrix} \delta & 0 & 0 \\ 0 & -\delta & 0 \\ 0 & 0 & \delta^2/(1-\delta^2) \end{bmatrix} \quad (5.16)$$

Second, a volume conserving orthorhombic strain (eq. 5.16) may be applied to the crystal [129]. The relation between the relaxed energy and that under strain is symmetric about $E(0)$ and is given by $E(\delta) = E(-\delta) = E(0) + (C_{11} - C_{12})V\delta^2 + O[\delta^4]$ [129]. V is the volume of the relaxed unit cell and $E(0)$ is the minimum energy. By fitting a polynomial to the energy-strain data, the coefficient for the quadratic term is equal to $(C_{11} - C_{12})V$.

$$\epsilon_{(C44)} = \begin{bmatrix} 0 & \frac{\delta}{2} & 0 \\ \frac{\delta}{2} & 0 & 0 \\ 0 & 0 & \delta^2/(4-\delta^2) \end{bmatrix} \quad (5.17)$$

Third, a volume conserving monoclinic strain (eq. 5.17) is applied and the relation between the relaxed energy, strained energy and the elastic constant C_{44} is $E(\delta) = E(-\delta) = E(0) + \frac{1}{2}C_{44}V\delta^2 + O[\delta^4]$. Similarly, fitting a polynomial to the strain-energy data points will calculate the value of C_{44} .

Finally, the relation between the bulk modulus, C_{11} and C_{12} , $B_0 = (C_{11} + 2C_{12})/3$ allows the computation of the individual constants eq. 5.18 (this relationship is only for cubic crystals).

$$\begin{aligned} C_{ortho} &= C_{11} - C_{12} \\ C_{11} &= \frac{3B_0 + 2C_{ortho}}{3} \\ C_{12} &= \frac{3B_0 - C_{ortho}}{3} \end{aligned} \quad (5.18)$$

5.2.6 Calculating Elastic Constants for Orthorhombic Crystal

The DFT work here includes Fe, Ru and Pd. Pure iron at room temperature is BCC, but this work is interested in austenitic stainless steel where the structure of atoms in the alloy are FCC. When modelling FCC iron using DFT with a non-polarized calculation, the crystal favours a cubic crystal with the atoms fixed in the FCC positions. When a spin-polarized calculation is computed, with magnetization along the z-axis, the crystal becomes slightly tetragonal with one side being 5% larger than the other two. This is a non-cubic crystal and the approach by Mehl et al to compute elastic constants is not applicable.

$$C_{ij} = \begin{bmatrix} C_{11} & C_{12} & C_{12} & 0 & 0 & 0 \\ C_{12} & C_{11} & C_{12} & 0 & 0 & 0 \\ C_{12} & C_{12} & C_{11} & 0 & 0 & 0 \\ 0 & 0 & 0 & C_{44} & 0 & 0 \\ 0 & 0 & 0 & 0 & C_{44} & 0 \\ 0 & 0 & 0 & 0 & 0 & C_{44} \end{bmatrix} \quad (5.19)$$

(3 independent values)

$$C_{ij} = \begin{bmatrix} C_{11} & C_{12} & C_{13} & 0 & 0 & 0 \\ C_{12} & C_{22} & C_{23} & 0 & 0 & 0 \\ C_{13} & C_{23} & C_{33} & 0 & 0 & 0 \\ 0 & 0 & 0 & C_{44} & 0 & 0 \\ 0 & 0 & 0 & 0 & C_{55} & 0 \\ 0 & 0 & 0 & 0 & 0 & C_{66} \end{bmatrix} \quad (5.20)$$

(9 independent values)

Once the relaxed crystal basis vectors and atom positions have been determined for the orthorhombic crystal, nine strains are applied to the crystal [130] in order to calculate the nine independent elastic constants.

As a strain is applied to a crystal its energy can change. The volume may also change, although it may be held constant. The relationship between the energy and the strain σ may be represented as a Taylor expansion of the internal energy in powers of the strain tensor[130]. The equation used by Ravindran et al is in a slightly different notation to that used by Mehl[131]. It includes a constant ξ to account for the symmetry of σ and the use of Voigt notation.

$$E(V, \sigma) = E(V_0, 0) + V_0 \left(\sum_i \tau_i \xi_i \sigma_i + \frac{1}{2} \sum_{ij} c_{ij} \sigma_i \xi_i \sigma_j \xi_j \right) + O(\sigma^3)$$

where σ is the strain
and τ is the stress
if the index k of ξ is $\in 1, 2, 3$ then $\xi_k = 1$
if the index k of ξ is $\in 4, 5, 6$ then $\xi_k = 2$

(5.21)

The first three strains applied to the orthorhombic crystal are non-volume conserving. They strain the crystal along each axis and allow the calculation of C_{11} , C_{22} and C_{33} .

$$D_1 = \begin{bmatrix} 1 + \delta & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (5.22) \qquad E(V, \sigma) = E(V_0, 0) + V_0 \left(\tau_1 \sigma + \frac{c_{11}}{2} \sigma^2 \right) \quad (5.23)$$

$$D_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 + \delta & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (5.24) \qquad E(V, \sigma) = E(V_0, 0) + V_0 \left(\tau_2 \sigma + \frac{c_{22}}{2} \sigma^2 \right) \quad (5.25)$$

$$D_3 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 + \delta \end{bmatrix} \quad (5.26) \qquad E(V, \sigma) = E(V_0, 0) + V_0 \left(\tau_3 \sigma + \frac{c_{33}}{2} \sigma^2 \right) \quad (5.27)$$

Volume conserving monoclinic distortions (eq. 5.28, eq. 5.30, eq. 5.32) are used to calculate the C_{44} , C_{55} and C_{66} elastic constants (eq. 5.29, eq. 5.31, eq. 5.33).

$$D_4 = \begin{bmatrix} \frac{1}{(1-\sigma^2)^{\frac{1}{3}}} & 0 & 0 \\ 0 & \frac{1}{(1-\sigma^2)^{\frac{1}{3}}} & \frac{\sigma}{(1-\sigma^2)^{\frac{1}{3}}} \\ 0 & \frac{\sigma}{(1-\sigma^2)^{\frac{1}{3}}} & \frac{1}{(1-\sigma^2)^{\frac{1}{3}}} \end{bmatrix} \quad E(V, \sigma) = E(V_0, 0) + V_0 \left(2\tau_4 \sigma + 2 \frac{c_{44}}{2} \sigma^2 \right) \quad (5.29)$$

(5.28)

$$D_5 = \begin{bmatrix} \frac{1}{(1-\sigma^2)^{\frac{1}{3}}} & 0 & \frac{\sigma}{(1-\sigma^2)^{\frac{1}{3}}} \\ 0 & \frac{1}{(1-\sigma^2)^{\frac{1}{3}}} & 0 \\ \frac{\sigma}{(1-\sigma^2)^{\frac{1}{3}}} & 0 & \frac{1}{(1-\sigma^2)^{\frac{1}{3}}} \end{bmatrix} \quad E(V, \sigma) = E(V_0, 0) + V_0 \left(2\tau_5 \sigma + 2 \frac{c_{55}}{2} \sigma^2 \right) \quad (5.31)$$

(5.30)

$$D_6 = \begin{bmatrix} \frac{1}{(1-\sigma^2)^{\frac{1}{3}}} & \frac{\sigma}{(1-\sigma^2)^{\frac{1}{3}}} & 0 \\ \frac{\sigma}{(1-\sigma^2)^{\frac{1}{3}}} & \frac{1}{(1-\sigma^2)^{\frac{1}{3}}} & 0 \\ 0 & 0 & \frac{1}{(1-\sigma^2)^{\frac{1}{3}}} \end{bmatrix} \quad E(V, \sigma) = E(V_0, 0) + V_0 \left(2\tau_6 \sigma + 2 \frac{c_{66}}{2} \sigma^2 \right) \quad (5.33)$$

(5.32)

Finally, three orthorhombic strains (eq. 5.34, eq. 5.36, eq. 5.38) that conserve the volume, by altering the strain along each axis, are used to calculate the C_{12} , C_{13} and C_{23} elastic constants (eq. 5.35, eq. 5.37, eq. 5.39).

$$D_7 = \begin{bmatrix} \frac{1+\sigma}{(1-\sigma^2)^{\frac{1}{3}}} & 0 & 0 \\ 0 & \frac{1-\sigma}{(1-\sigma^2)^{\frac{1}{3}}} & 0 \\ 0 & 0 & \frac{1}{(1-\sigma^2)^{\frac{1}{3}}} \end{bmatrix} \quad E(V, \sigma) = E(V_0, 0) + V_0 \left((\tau_1 - \tau_2 \sigma + \frac{1}{2}(c_{11} + c_{22} - 2c_{12})\sigma^2) \right) \quad (5.35)$$

(5.34)

$$D_8 = \begin{bmatrix} \frac{1+\sigma}{(1-\sigma^2)^{\frac{1}{3}}} & 0 & 0 \\ 0 & \frac{1}{(1-\sigma^2)^{\frac{1}{3}}} & 0 \\ 0 & 0 & \frac{1-\sigma}{(1-\sigma^2)^{\frac{1}{3}}} \end{bmatrix} \quad E(V, \sigma) = E(V_0, 0) + V_0 \left((\tau_1 - \tau_3 \sigma + \frac{1}{2}(c_{11} + c_{33} - 2c_{13})\sigma^2) \right) \quad (5.37)$$

(5.36)

$$D_9 = \begin{bmatrix} \frac{1}{(1-\sigma^2)^{\frac{1}{3}}} & 0 & 0 \\ 0 & \frac{1+\sigma}{(1-\sigma^2)^{\frac{1}{3}}} & 0 \\ 0 & 0 & \frac{1-\sigma}{(1-\sigma^2)^{\frac{1}{3}}} \end{bmatrix} \quad E(V, \sigma) = E(V_0, 0) + V_0 \left((\tau_2 - \tau_3 \sigma + \frac{1}{2}(c_{22} + c_{33} - 2c_{23})\sigma^2) \right) \quad (5.39)$$

(5.38)

5.2.7 Stability Conditions and Other Properties

Once the elastic constants are calculated, a number of other properties may be computed from them. The stability of a crystal, bulk modulus, shear modulus and Young's modulus may all be found from the elastic constants. There is also a correlation between these and the melting temperature of a material. These are discussed in detail in appendix I and were implemented in the potential fitting code that results from this work.

5.3 Interatomic Potentials and Choice of Function

5.3.1 Introduction

An interatomic potential, as used in this work and MD computer codes, is a function or a set of functions that describe the energy and force between atoms. Simpler functions represent the potential energy between pairs of atoms only, but more complicated functions have been used in molecular dynamics since the 1980s that also attempt to represent the many body nature of materials, which applies in particular to metals.

5.3.2 Pair Potentials

A pair potential only considers individual pairs of nearby atoms, one pair at a time, and does not consider the effect of any other nearby atoms. Where an alloy is being modelled, there will be a pair potential function for each element and element combination; 1 for a single element, 3 for a two element alloy, 6 for a three element alloy and so on ($n(n + 1)/2$).

Lennard-Jones

The Lennard-Jones potential (eq. 5.40)[132], fig. 5.3, was proposed in the 1920s and has both an attractive term and a repulsive term; the $(r_m/r)^{12}$ becomes much larger than the $(r_m/r)^6$ term at close distances, and this mimics the coulomb repulsion as two atoms are pushed closer together. At larger separations, the attractive term dominates.

$$V(r) = e \left(\left(\frac{r_m}{r} \right)^{12} - 2 \left(\frac{r_m}{r} \right)^6 \right) \quad (5.40)$$

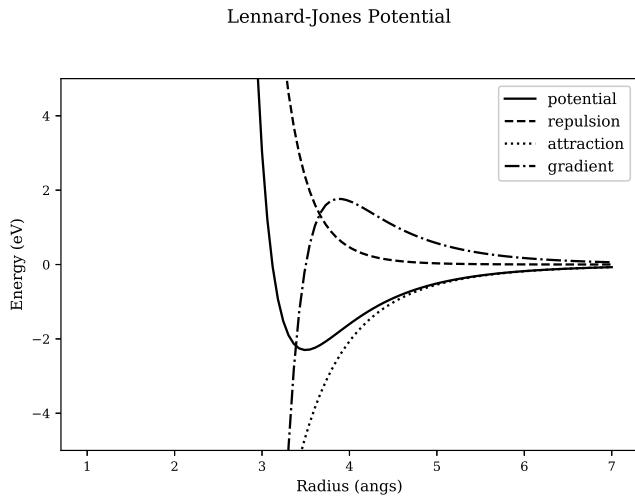


Figure 5.3: Lennard-Jones Potential

The Lennard-Jones potential ($V(r)$) requires two parameters (e, r_m) and is dependent upon the atom separation (r).

Morse Potential

The Morse potential (eq. 5.41)[133], fig. 5.4, was proposed five years after the Lennard-Jones potential. It also has an attractive and repulsive term, but it uses the exponential function rather than 6th and 12th powers.

$$V(r) = \exp(-2a(r - re)) - 2 \exp(-a(r - re)) \quad (5.41)$$

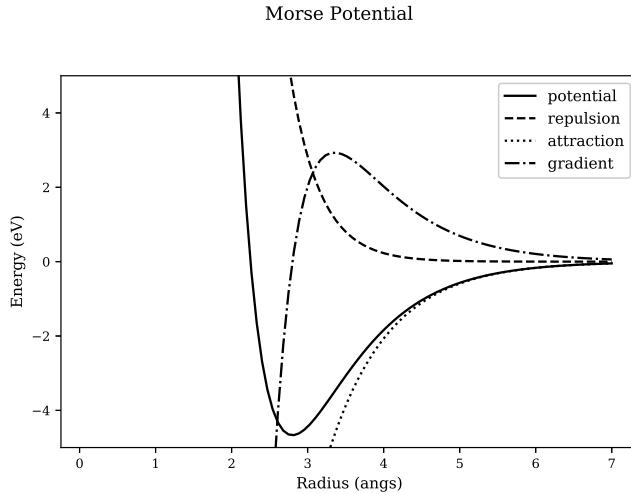


Figure 5.4: Morse Potential

The Morse potential ($V(r)$) requires two parameters (a, r_e) and is dependent upon the atom separation (r).

Buckingham Potential

The Buckingham Potential (eq. 5.42)[134] consists of a repulsive and attractive part. As the separation decreases, the attractive term dominates and the overall function tends towards negative infinity. This shouldn't be problematic when gasses or solids are being modelled alone, but when collisions and damage cascades are being modelled, the separation between atoms may be much smaller than that in a typical simulation, ending with these atoms being pulled together (fig. 5.5).

$$V(r) = A \exp(-Br) - \frac{C}{r^6} \quad (5.42)$$

The Buckingham potential ($V(r)$) requires three parameters (A, B, C) and is dependent upon the atom separation (r).

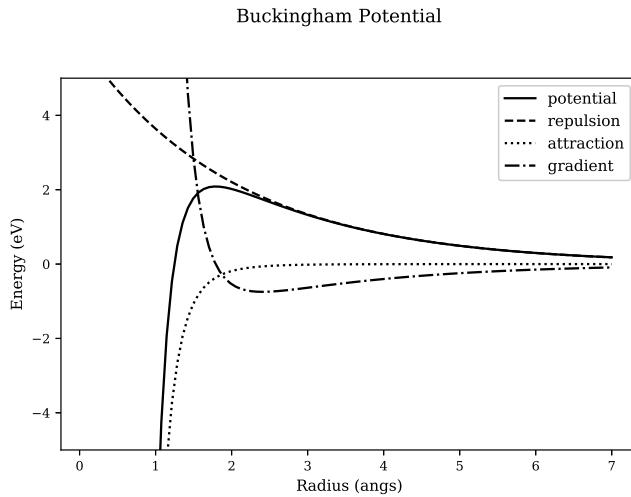


Figure 5.5: Buckingham Potential

Ziegler-Biersack-Littmark Universal Potential Function

It became clear, while experimenting with a number of existing potentials and MD programs, that at the very least modifications to those potentials would need to be made for small atom separations. A simulation to model a projectile failed early on due to the projectile's proximity to target atoms, resulting in the MD code returning an error as the separation was out of the range of the potential.

The Ziegler-Biersack-Littmark (eq. 5.43) potential (fig. 5.6), between an atom of charge Z_i and Z_j is set out in the SRIM book[118].

$$V_{zbl}(r_{ij}, Z_i, Z_j) = \frac{1}{4\pi\epsilon_0} \frac{Z_i Z_j}{r_{ij}} \phi\left(\frac{r_{ij}}{a_{ij}}\right) \quad (5.43)$$

where $\epsilon_0 = 8.85419 \times 10^{-12}$

The parameters of the Universal screening potential $\phi\left(\frac{r_{ij}}{a_{ij}}\right)$ function are as follows:

$$\phi(x) = 0.181 \exp(-3.2x) + 0.5099 \exp(-0.9423x) + 0.2802 \exp(-0.4028x) + 0.02817 \exp(-0.2016x)$$

where $a_{ij} = \frac{0.8854a_0}{Z_i^{\frac{2}{3}} + Z_j^{\frac{2}{3}}}$ (5.44)

and $a_0 = 0.529$ angstrom (Bohr radius)

The Universal screening function has 8 parameters defined. They do vary depending on the source, but these are as defined in the SRIM 2015 book[118]. The argument x is equal to $\frac{r_{ij}}{a_{ij}}$ where r_{ij} is the separation between atoms and a_{ij} is the screening length defined by J. F. Ziegler, J. P. Biersack and M. D. Ziegler[118].

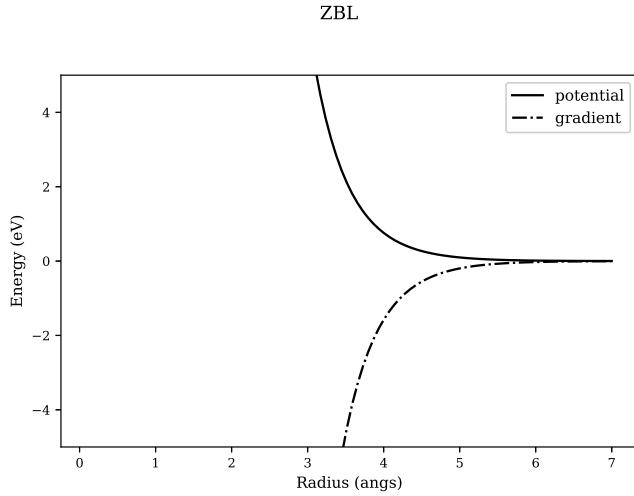


Figure 5.6: ZBL universal screening potential

5.3.3 Many Body Potentials

This work is focused on metals, and two popular, and closely related potentials, will be discussed in this section.

Finnis-Sinclair

The Finnis-Sinclair potential was published in 1984[135] and it introduced both a pair potential and an embedded term to take into account the cohesive energy dependent on the local electron density. The pair term represents the repulsion between the atoms whereas the embedding functional glues the atoms together in the solid.

$$U_{total} = U_P + U_N \quad (5.45)$$

The tight-binding that this potential is based on is represented by the functional: square root of the density. The potential for a single element model consists of a pair function (eq. 5.46), a density function and a tight-binding functional (eq. 5.47).

$$U_P = \frac{1}{2} \sum_{ij} V(R_{ij}) \quad (5.46)$$

$$\begin{aligned} U_N &= -A \sum_i f(\rho_i) \\ p_i &= \sum_j \phi(R_{ij}) \end{aligned} \quad (5.47)$$

$$R_{ij} = |\vec{R}_j - \vec{R}_i| \quad (5.48)$$

The total energy (U_{total}) is the sum of the total pair (U_P) and total N-body terms (U_N). Both the pair

potential ($V(R_{ij})$) and electron density ($\phi(R_{ij})$) are dependent upon the atom separation distance (eq. 5.48). In the Finnis-Sinclair model, the embedded functional ($f(\rho_i)$) is chosen to be $\sqrt{\rho}$ in order to mimic tight-binding theory[135]. This functional is multiplied by a parameter (A), a value determined by the author of the potential.

Embedded-atom method

The embedded-atom method was also invented in the 1980s by Daw and Baskes[136] (eq. 5.49). It was developed with metals in mind, and in many ways is a more flexible variant of the Finnis-Sinclair potential. It too has a pair and density function, but the functional of the EAM potential is not restricted to a square root.

$$U_{EAM} = \frac{1}{2} \sum_{i=1}^N \sum_{j \neq i}^N V_{ij}(r_{ij}) + \sum_{i=1}^N F[\rho_i] \quad (5.49)$$

where $\rho_i = \sum_{j=i, j \neq i}^N \rho_{ij}(r_{ij})$

Elastic properties were not reliably calculated from pair functions alone[136], but the advent of Finnis-Sinclair and EAM type potentials changed this.

$$A_{ij} + F'(\vec{\rho})V_{ij} = 0$$

where

$$A_{ij} = \frac{1}{2} \sum_m \phi'_m a_i^m a_j^m / a^m \quad (5.50)$$

$$V_{ij} = \frac{1}{2} \sum_m \rho'_m a_i^m a_j^m / a^m$$

For homonuclear cubic crystals, the lattice constant may be calculated by the equilibrium condition in eq 5.50. The three independent elastic constants, C_{11}, C_{12}, C_{44} , may also be calculated at equilibrium from eq 5.51.

$$C_{11} = (B_{11} + F'(\vec{\rho})W_{11} + F''(\vec{\rho})(V_{11})^2) / \Omega_0$$

$$C_{12} = (B_{12} + F'(\vec{\rho})W_{12} + F''(\vec{\rho})(V_{11})^2) / \Omega_0$$

$$C_{44} = (B_{12} + F'(\vec{\rho})W_{12}) / \Omega_0$$

where

$$B_{ijkl} = \frac{1}{2} \sum_m (\phi''_m - \phi'_m / a^m) a_i^m a_j^m a_k^m a_l^m / (a^m)^2 \quad (5.51)$$

$$W_{ijkl} = \frac{1}{2} \sum_m (\rho''_m - \rho'_m / a^m) a_i^m a_j^m a_k^m a_l^m / (a^m)^2$$

where

$$\phi''_m = (d^2\phi(r)/dr^2)_{r=a^m},$$

$$\rho''_m = (d^2\rho(r)/dr^2)_{r=a^m}.$$

There are notable consequences in removing either the pair function or embedding functional. If $V_{ij}(r_{ij})$ is removed, $F'[\vec{\rho}] = 0$ and this gives a shear modulus of 0 ($C_{11} = C_{12}, C_{44} = 0$)[136]. If $F[\vec{\rho}]$ is removed the Cauchy discrepancy becomes zero ($C_{12} = C_{44}$)[136]. Both of these situations are generally untrue, and both functions are required for a good potential for a metal.

Two band embedded-atom method

There are several variations of the potential, and one of particular interest to us is the two band embedded-atom method that has two electron density and embedding energy terms. This formalism was originally developed to model Cs[137], and the transition of electrons between S and D bands under pressure, but it has been modified to apply to alloys.

Cs changes its structure as pressure is applied. The first change is from BCC to a more compact FCC structure, but it then compresses further as electrons move from the S-band to the more compact D-band, reducing the size of the atom.

The bond energy may be described with eq. 5.52 where N_1 and N_2 are the capacities of each band, n_{i1} and n_{i2} are the electrons in each band of the i^{th} atom. E_{prom} is the energy required to promote an electron from band 1 to band 2.

$$U_{\text{bond}} = \sum_i \frac{W_{i1}}{2N_1} n_{i1}(n_{i1} - N_1) + \sum_i \frac{W_{i2}}{2N_2} n_{i2}(n_{i2} - N_2) + E_{\text{prom}} \quad (5.52)$$

A Finnis-Sinclair type EAM potential was used by Ackland and Reed in this work for both bands. The parameters fitted in Ackland and Reed's work are listed in table 5.3.

$$\begin{aligned} & \text{Pair} \\ V_s(r_{ij}) &= \sum_i \frac{A_s}{r_{ij}^{12}} \\ V_d(r_{ij}) &= \sum_i \frac{A_d}{r_{ij}^{12}} \end{aligned} \quad (5.53)$$

$$\rho_s(r_{ij}) = \begin{cases} C_s(d_s - r_{ij})^3 & r_{ij} < d_s \\ 0 & r_{ij} > d_s \end{cases} \quad \rho_d(r_{ij}) = \begin{cases} C_d(d_d - r_{ij})^3 & r_{ij} < d_d \\ 0 & r_{ij} > d_d \end{cases} \quad (5.54)$$

$$\text{Embedding} \quad (5.55)$$

Parameter	Value
C_s	$0.05617 \text{ eV}^2 \text{ ang}^{-3}$
C_d	$0.1681 \text{ eV}^2 \text{ ang}^{-3}$
d_s	9.5097 ang
d_d	6.9189 ang
A_s	$2.417 \times 10^7 \text{ ang}^{12}$
A_d	$3.7668 \times 10^6 \text{ ang}^{12}$

Table 5.3: Caesium 2BEAM parameters

The implementation of this two-band potential predicted the transformation of Cs. The I phase, at ambient pressure, is BCC Cs with a primitive cell volume of 115.9 cubic angstrom per atom. As more pressure is added, the optimum structure changes from BCC to FCC, and this results in a smaller volume per atom of 67.5 cubic

angstrom. Finally Cs undergoes an isostructural transformation, and the potential predicts a volume of 48.7 cubic angstrom per atom.

There is a transition pressure of 4.3 GPa between the phases II and III. The potentials developed by Ackland and Reed were “in good agreement with experiment”[137].

An alloy version of the two-band model was developed by Olsson et al to investigate the α -prime phase formation in Fe-Cr[138]. Fe-Cr alloys form ferromagnetic metals with concentrations of up to 10% chromium at $750^{\circ}C$. As the concentration of chromium in the alloy increases, the alloy begins to decompose into iron rich and chromium rich precipitates, and this decomposition is accelerated under irradiation.

The two-band method for Cs was extended to an Fe-Cr alloy in order to describe the heat of mixing in the alloy.

$$U_{EAM} = \frac{1}{2} \sum_{i=1}^N \sum_{j \neq i}^N V_{ij}(r_{ij}) + \sum_{i=1}^N F_D[\rho_{d,i}] + \sum_{i=1}^N F_S[\rho_{s,i}] \quad (5.56)$$

where $\rho_{d,i} = \sum_{j=i,j \neq i}^N \rho_{d,ij}(r_{ij})$ and $\rho_{s,i} = \sum_{j=i,j \neq i}^N \rho_{s,ij}(r_{ij})$

The embedding functional (eq. 5.60) was in the form of several functionals used in papers by Ackland and Mendelev, and an extension to the standard Finnis-Sinclair.

$$F_{band}(\rho_{band}) = A_1^{band} \sqrt{\rho_{band}} + A_2^{band} \rho_{band}^2 + A_3^{band} \rho_{band}^4 \quad (5.57)$$

The choice of function for both the pair and d-band density functions was a cubic spline, and a 4s Slater type function was used for the s-band alloy density. The Fe-Fe and Cr-Cr s-band density functions were zero functions. Where the alloy has high concentrations of Iron or Chromium, the respective d-band density functions will dominate, but as the elements mix, the s-band density will also contribute.

$$V(r) = \sum_i a_i (r - r_i)^3 H(r_i - r) \text{ where } H \text{ is the Heaviside Step function} \quad (5.58)$$

$$\phi_d^{CrCr}(r) = \sum_k b_k (r - r_k)^3 H(r_k - r) \text{ where } H \text{ is the Heaviside Step function} \quad (5.59)$$

$$\begin{aligned} \phi_s^{FeCr}(r) &= (H_s r^3 \exp(-\xi_s r))^2 \\ \phi_s^{FeFe}(r) &= 0.0 \\ \phi_s^{CrCr}(r) &= 0.0 \end{aligned} \quad (5.60)$$

The resulting potential, with the s-density fitted to the mixing enthalpy of Fe and Cr, reproduces the formation of α -prime phase Cr with thermal ageing over a range of Cr concentrations.

The Fe-Cr potentials were further developed by Bonny et al. The mixing enthalpy changes sign in Fe-Cr, having a positive mixing enthalpy for Cr concentrations above 10% and a negative mixing enthalpy below. This results in the solubility of Cr in the alloy at lower concentrations, but as the concentration rises there’s a tendency for Cr to form α -prime precipitates[139].

In the work by Bonny et al, the base Iron and Chromium potentials were those developed by Mendelev and Ackland, and from the Fe-Cr potentials of Olsson and Wallenius. In total, 11 functions make up the Bonny et al potential, higher than the usual 7 functions required for a two species EAM. In a standard EAM potential, the density contribution from an atom of one species is only dependent on that contributing atom, and not the embedded atom. This potential splits the density function into multiple bands and multiple permutations of atom species.

$$\begin{aligned}\rho_{AA}^d(r) &= \rho_{BA}^d = \rho_A^d \\ \rho_{BB}^d(r) &= \rho_{AB}^d = \rho_B^d \\ \rho_{AB}^s(r) &= \rho_{BA}^s \\ \rho_{AA}^s(r) &= \rho_{BB}^s = 0\end{aligned}\tag{5.61}$$

The chosen s-band density function for the Bonny et al Fe-Cr alloy was selected as an exponential style function with a cut-off function.

$$\rho_{FeCr}^s(r) = kr^6 \exp(-2\xi r)g_{cut}(r)\tag{5.62}$$

$$g_{cut}(r) = \begin{cases} 1 & r \leq r_c^i \\ \frac{1}{2} \left(1 - \sin\left(\frac{\pi}{2} \frac{r-r_m}{d}\right)\right) & r_c^i < r < r_c^f \\ 0 & r_c^f \leq r \end{cases}\tag{5.63}$$

$$F^s(\rho) = A_1\sqrt{\rho} + A_2\rho^2\tag{5.64}$$

The magnetic properties of Cr were considered during the development of these potentials. First-Principles calculations are equivalent to calculating properties at 0K, and at this temperature Cr is antiferromagnetic. The Neel temperature for Cr, the point at which it transitions from an antiferromagnetic to a paramagnet, is 310K. The operating temperature of the alloys, within a reactor, will be above the Neel temperature. Cr has a positive Cauchy pressure as a paramagnet, and a negative Cauchy pressure under the Neel temperature when an antiferromagnetic, and as a result of this and the operating temperature, a positive Cauchy pressure was used to fit the potential.

Modified embedded-atom method

All the potential types considered so far are spherically symmetrical about the atom. Typically covalent bonds tend to be directional but metallic bonds, in contrast, are isotropic about each atom. However, in alloys and magnetic materials where the system is not isotropic, the modified embedded atom method may be used.

$$U_{MEAM,i} = \frac{1}{2} \sum_{j \neq i}^N V_{ij}(r_{ij}) + F_i(\vec{\rho}_i)\tag{5.65}$$

The original form for the embedding functional is given below (eq. 5.66)[140].

$$F(\vec{\rho}) = AE_{coh} \frac{\vec{\rho}}{\vec{\rho}^0} \ln \left(\frac{\vec{\rho}}{\vec{\rho}^0} \right) \quad (5.66)$$

The electron density is made from four contributing functions. The first, ρ^0 , is not dependent on direction, but ρ^1 , ρ^2 and ρ^3 are dependent on the angle.

$$\begin{aligned} \rho^0 &= \sum_i^N \rho_i^0(r^i) \\ (\rho^1)^2 &= \sum_{\alpha} \left(\sum_i \rho_i^1(r^i) \frac{r_{\alpha}^i}{r^i} \right)^2 \\ (\rho^2)^2 &= \sum_{\alpha} \left(\sum_i \rho_i^2(r^i) \frac{r_{\alpha}^i r_{\beta}^i}{r^i r^i} \right)^2 - \frac{1}{3} \left(\sum_i \rho_i^2(r^i) \right)^2 \\ (\rho^3)^2 &= \sum_{\alpha} \left(\sum_i \rho_i^2(r^i) \frac{r_{\alpha}^i r_{\beta}^i r_{\gamma}^i}{r^i r^i r^i} \right)^2 - \frac{3}{5} \left(\sum_{\alpha} \frac{r_{\alpha}^i}{r^i} \rho_i^3(r^i) \right) \end{aligned} \quad (5.67)$$

5.3.4 EAM and Two-Band EAM Functions

Previous work has shown there to be an extensive set of functions used to represent the pair potential, density and embedding functional. These range from those that preserve and attempt to replicate the physics, to those that do away with knowledge of the physics underlying the potentials altogether. The pair potentials already covered here may also be used as the pair function component of the EAM or 2BEAM potentials. A full list of the functions considered and included in the potential fitting code in this work is included in appendix T.

Tending to Zero at the Cutoff Radius

It is desirable for the functions to be well behaved and continuous. The coulomb force between two charged particles reduces smoothly until theoretically at an infinite separation; it doesn't reach a set separation and abruptly drop to zero. It is impossible and unhelpful to consider a very large or infinite separation which is why the cutoff radius has been introduced. It represents the separation where the potential has reached zero; at the very least it's a trade off between accuracy and the computing time available for the problem.

The pair potential should therefore drop off to 0 at the cut off radius, and be equal to zero for larger separations. The electron density spherically around an atom should also smoothly drop off to zero, and similarly it should be equal to zero for distances larger than the cut off. The embedding energy is dependent on the density at the location of the atom embedded. The embedding function will not necessarily drop off to zero, and it depends on the density and not the separation.

In a MD simulation, the neighbour list will usually be built considering atoms slightly more separated than the cut off radius of the functions. This allows the same neighbour list to be used for several time steps, before the need to update as atoms will get closer and further apart with each time step.

Collisions and Interatomic Potentials

In a simulation at room temperature the average energy of the atoms in the simulation are approximately 0.05eV, and at temperatures near the melting point of iron the energy is approximately 0.2eV. These are energy

ranges that interatomic potentials are designed to operate in, but in typical damage cascades in nuclear core components the atoms in the cascade, including the PKAs, have energies up to approximately 100keV.

If a potential is not designed for use in collision simulations, it may not contain the data points required for such close separations. If a potential, such as the Buckingham potential (section 5.3.2), is used, the energy peaks at a certain separation and then drops once more. This would cause colliding atoms to stick to one another once they reached a certain separation.

The ZBL function is designed specifically with collisions in mind and should be used in MD simulations that involve collisions. In simulations of BCC tungsten[141], the material had an initial temperature of 10K and 523K with PKAs energies at 10keV, 20keV and 50keV. The models were simulated with MOLDY using an EAM potential where the pair potential is splined to a ZBL for short range interactions.

5.4 EAM Potential Fitting

5.4.1 Sacrificing Physical Elegance

We have no knowledge of a formula that can be applied to any material and describe the energy and forces between atoms exactly. Even first principles calculations rely on approximations, and the knowledge of an exact exchange functional eludes us. While it would be satisfying to derive a potential that reflects the fundamental physics, it may be something that is forever out of our reach.

It is more useful to develop a potential that replicates the properties of a material such as the relaxed lattice parameter, cohesive energy and elastic constants. The ZBL potential is an approximation to the repulsion between ions and is an appropriate potential to use to model high energy collisions, but not the bulk material. The Finnis-Sinclair are rooted in the second moment approximation of tight-binding theory and has been used to model metals.

Alternative potentials functions that lose physical significance may also be used to give potentials that work for specific elements under certain conditions. The trade-off for this loss any physical elegance [137] is a potential that replicates the behaviour of a material under the conditions that are being studied, rather than use a more generic potential that isn't detailed enough to do so.

5.4.2 Exact Fitting to Parameters

As discussed in section 5.3.3 the functions may be fit directly to the known parameters for the metal. This was also the approach taken by Mishin whilst developing an alloy potential to model an Ni-Al system[142].

An exact fitting code was developed by G. Bonny in order to fit cubic spline type functions exactly to relaxed data including the Rose-Vinet equation of state (see section 5.2.2). Following work by Ackland[137], Mendelev[143], Olsson and Wallenius[138] the code developed by Bonny et al was used to develop a concentration dependent potential for Iron-Chromium[128]. More recently a potential has been fit for Tungsten-Rhenium alloys[126]. This code uses a quadratic program[144] to fit the potentials to the desired properties.

5.4.3 Force Matching

Force data, gathered experimentally or by first-principles calculations, has been used to develop potentials since the 1990s. The force matching method was developed in 1994 by Ercolessi and Adams [145] to link the more accurate, more processor and memory intensive, world of first-principles calculations to Molecular Dynamics. The force-matching method uses the difference between the actual force, typically calculated by first-principles calculations, and that given by the potential.

With a set of M different atomic configurations, and a potential with a set of L parameters (\vec{p}), the function Z_F is a measure of the difference between the forces calculated using the potential for all configurations and the actual (or DFT generated) forces.

$$Z_F(\vec{\alpha}) = \sum_{k=1}^M \sum_{i=1}^k \sum_{j=1}^3 |F_{i,j}^k(\vec{\alpha}) - F_{i,j}^0|^2 \quad (5.68)$$

This may be extended to include the calculation of other properties:

- Lattice Parameter a_0
- Cohesive Energy E_{coh}
- Bulk Modulus B_0
- Equation of State (E_0, V_0, B_0, B_0^p)
- Stress Tensor
- Elastic Constants
- Shear Modulus, Young's Modulus, Poisson Ratio
- Melting Temperature
- Surface Energy

Each may be weighted depending on how important the property is to the simulation the potential is required for.

$$Z(\vec{p}) = w_F Z_F(\vec{p}) + w_{b0} Z_{b0}(\vec{p}) + w_{e0} Z_{e0}(\vec{p}) + w_{a0} Z_{a0}(\vec{p}) + w_{ec} Z_{ec}(\vec{p}) + w_{coh} Z_{coh}(\vec{p}) \quad (5.69)$$

The result is a residual squared sum that measures how well or poorly a potential fits the required data.

5.4.4 Pair and Embedding Symmetry Transforms

The combination of density function and embedding functional are not unique. When a potential is developed for a single element, this isn't a problem, but it must be considered when fitting to an alloy. If the density is based on real data, such as density distribution computed using a first principles code such as DIRAC, then there is some underlying meaning to the function. However, the functions fitted in this work have no real resemblance to the physics, other than having an effective cut off.

Bonny and Malerba[146] show two transforms that may be applied to the pair function, density function and embedding functional that the energy of any given atom is invariant to.

$$E_\alpha = \frac{1}{2} \sum_{\beta \neq \alpha} V(R_{\alpha\beta}) + F[\bar{\rho}_\alpha] \text{ where } \bar{\rho}_\alpha = \sum_{\beta \neq \alpha} \rho(R_{\alpha\beta}) \quad (5.70)$$

Given the usual definition of the EAM, in eq. 5.70, the first transform scales the value of the density function as simultaneously, and inversely, with the argument of the embedding functional (eq. 5.71). The second transform applies to the pair function and embedding functional (eq. 5.72).

$$\tilde{\rho}(R_{\alpha\beta}) = S\rho(R_{\alpha\beta})\tilde{F}[\bar{\rho}_\alpha] = F[\bar{\rho}_\alpha/S] \quad (5.71)$$

$$\tilde{F}[\bar{\rho}_\alpha] = F[\bar{\rho}_\alpha] + C\bar{\rho}_\alpha\tilde{V}(R_{\alpha\beta}) = V(R_{\alpha\beta}) - 2C\rho(R_{\alpha\beta}) \quad (5.72)$$

Mishin's approach, when fitting a binary alloy of Ni and Al, was to identify three transforms with three parameters that could be adjusted whilst fitting the alloy potential (eq. 5.73).

$$\begin{aligned} \tilde{\rho}_B &= s_B\rho_B(r) \quad \tilde{F}_B(\bar{\rho}) = F_B(\bar{\rho}/s_B) \\ \tilde{F}_A(\bar{\rho}) &= F_A(\bar{\rho}) + g_A\bar{\rho} \quad \tilde{V}_{AA}(r) = V_{AA}(r) - 2g_A\rho_A(r) \\ \tilde{F}_B(\bar{\rho}) &= F_B(\bar{\rho}) + g_B\bar{\rho} \quad \tilde{V}_{BB}(r) = V_{BB}(r) - 2g_B\rho_B(r) \end{aligned} \quad (5.73)$$

Bonny and Malerba chose to apply an effective gauge to the potentials for pure species. The gauge they chose sets the gradient of the embedding functional to zero at the equilibrium density, that of the relaxed crystal, as well as setting the equilibrium density to zero.

To transform the embedding functional such that $F'[\bar{\rho}_{eq}] = 0$, the constant C is set as $C = -F'[\bar{\rho}_{eq}]$. To transform the density such that $\bar{\rho}_{eq} = 1$ the constant S is set as $S = 1/\bar{\rho}_{eq}$ [146].

In the case of a binary alloy, the potentials for the pure elements are derived. Following this, the effective gauge transforms are applied to both pure element potentials. Finally, the cross pair potential is fit whilst the pure potentials are fixed.

5.5 Classical Molecular Dynamics

5.5.1 Introduction

Ab initio (DFT) calculations approximately solve the Schrödinger equation to calculate the energy, forces and stress of a given simulation. They take a relatively long time to calculate, have relatively small numbers of atoms (hundreds to thousands) and are fixed at one moment in time, although there are now programs that will run DFT MD. In comparison, MD models a collection of atoms over a specified period of time. Much larger collections of atoms are possible (thousands to millions) and the interaction between atoms are predefined by interatomic potentials.

Molecular dynamics has been used to investigate the effects of radiation on materials. The damage event on the atomic scale is rapid in time, in the femtosecond to picosecond range, and affects a small volume of the material whereas the effects of the damage to the material on a mesoscopic and macroscopic scale may range from days to decades. In reactor pressure vessels, for instance, the damage and defect production will cover a large range of time and size scales[31]:

- $10^{-10}m$ to $10^{-3}m$ (10^{18} to 10^{25} atoms)
- $10^{-17}s$ to 10^9s (tens of attoseconds to 30 years)

As a major element of steel, iron has often been the subject of MD simulations. Damage cascades have been studied in iron at a range of temperatures and PKA energies.

The development of new potentials coupled with both MD and other tools, such as akMC, allows the study of radiation damage in a way that is not possible experimentally. While a full MD simulation of a material over many years is beyond current technology, many useful properties may be extracted. This could include vacancy energy barriers, energetics of a system modelling grain boundaries, the formation of defects and prismatic loops.

5.5.2 Molecular Dynamics Codes

Many MD codes are available for researchers to download freely and use from the Internet (although, terms may apply to the use). These include DL_POLY (Fortran), Large-scale Atomic/Molecular Massively Parallel Simulator (LAMMPS) (C++) and MOLDY (C). Most are able to run using Message Parsing Interface (MPI), Open Multi-Processing (OpenMP) or a hybrid to leverage executing the program in parallel across many compute nodes. Versions of these codes also exist to take advantage of the massive parallelism that exists in graphical processing units.

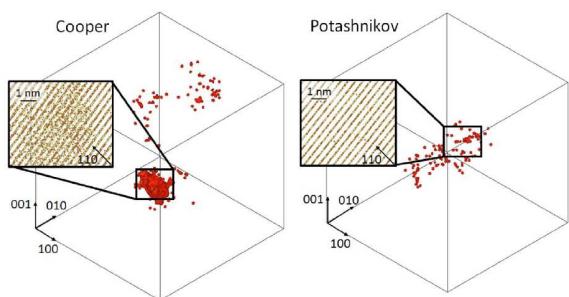


Figure 5.7: Damage in MOX (LAMMPS)[147]



Figure 5.8: Sputtering of plutonium oxide (LAMMPS)[148]

Damage cascades in mixed oxides have been modelled recently using the LAMMPS code. A 70x70x70 box with sides approximately 40nm in length and containing 4,116,000 atoms was used with PKAs having energies ranging from 5 to 75keV[147] (fig. 5.7). The sputtering of material from the surface of plutonium oxide (IV) has also been modelled with LAMMPS using a box of 393,216 atoms with a PKA of 87.7keV[148] (fig. 5.8). The timescale of the sputtering simulation was 7.8 picoseconds (7.8×10^{-12}) and the entire event was split into 1.0×10^5 time steps of 7.8×10^{-17} s.

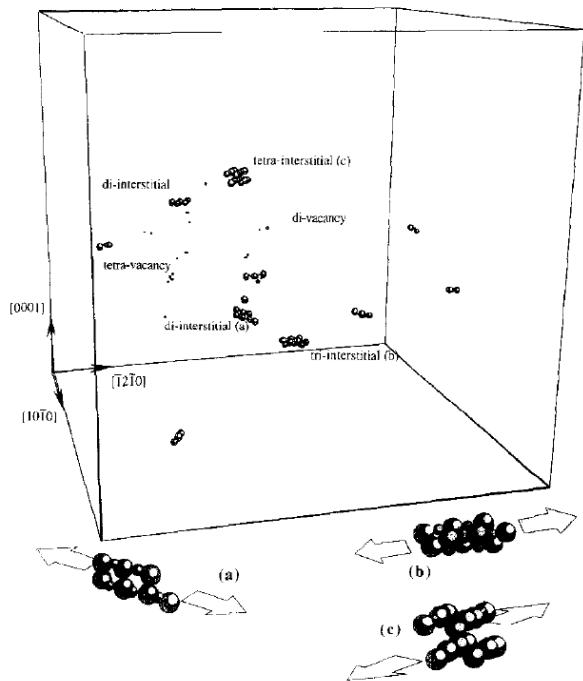


Figure 5.9: 5keV cascade (MOLDY)[149]

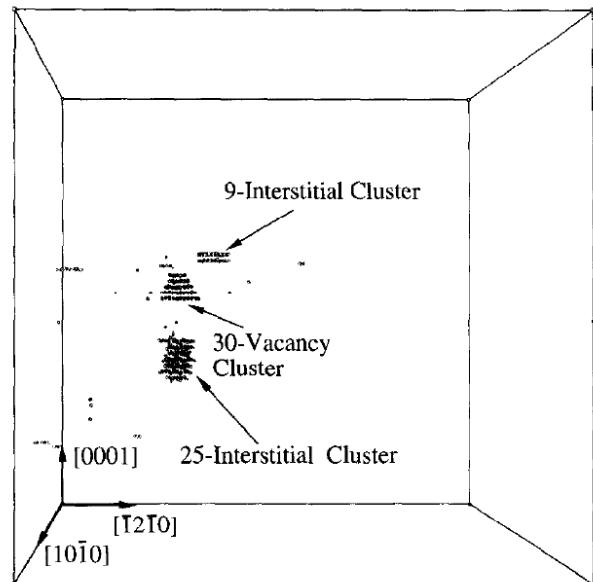
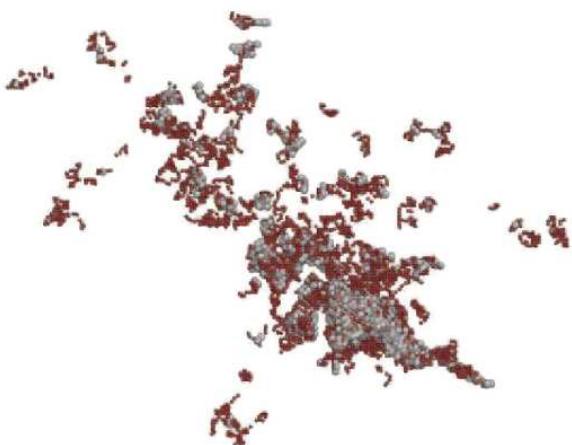


Figure 5.10: 20keV cascade (MOLDY)[149]

MOLDY has previously been used to model primary knock on atoms in metals, at energies of 5kev, 10keV and 20keV which would be slightly lower than the typical iron PKAs close to U235 fuel in the reactor (fig. 1.8). This simulation however was in alpha-zirconium and the higher mass of zirconium, relative to iron, would lead to lower energy PKAs. The damage cascades were modelled in a 104,832 atom block for the 5keV PKA (fig. 5.9) and a 445,536 atom block for the 20keV PKA (fig. 5.10).

Figure 5.11: 100keV cascade after 50ps in TiO_2 (DL_POLY)[150]

DL_POLY has also been used to model damage cascades in various oxides (titanium, aluminium, silicon)[150]. The PKA energies are higher than those performed in MOLDY 15 years earlier, but still under 1MeV. The authors expect 1MeV simulation boxes to require at least 1 billion atoms, but the 100-200keV PKAs they modelled were within 10 million atom simulation boxes. Slightly larger boxes and damage cascades required longer simulation times, and with a box size of 43nm per side, a 50ps simulation time was used (fig. 5.11).

MD codes compute the energy, forces and stress of a collection of atoms in similar ways, although code optimizations may vary. An overview of the methods used, and replicated in the code created for this work, are discussed in appendix R.

5.5.3 Velocity Verlet Algorithm

Verlet integration is used by MD codes such as DL_POLY, LAMMPS, Democritus as well as MD DFT codes, as in the case of CASTEP.

First, the forces at time step n are calculated \vec{f}_n . Now, using these forces and the velocity at time step n, the half step velocity is computed (eq. 5.74).

$$\vec{v}_{n+\frac{1}{2}} = \vec{v}_n + \frac{\vec{f}_n \delta t}{2m} \quad (5.74)$$

The position of the next step, n + 1, is calculated using the position at n, the time step δt and the half step velocity (eq. 5.75).

$$\vec{r}_{n+1} = \vec{r}_n + \vec{v}_{n+\frac{1}{2}} \times \delta t \quad (5.75)$$

The forces are recalculated at the new position \vec{r}_{n+1} to give \vec{f}_{n+1} and this, along with the half step velocity, is used to compute the velocity at step n+1, \vec{v}_{n+1} (eq. 5.76).

$$\vec{v}_{n+1} = \vec{v}_{n+\frac{1}{2}} + \frac{\vec{f}_{n+1} \delta t}{2m} \quad (5.76)$$

This process is repeated for each time step δt from the start of the simulation to the end.

5.6 Existing Potential Fitting Programs

Bonny Quadratic Program Fitting

This code has been used to fit a number of potentials, including Fe-Cr and W-Re. It is a Fortran code and uses known properties of the material in different configurations. The only functional forms available in the 2012 version of the code are cubic spline for the pair and density functions and a polynomial for the embedding functional. The fitting subroutine uses quadratic programming to find parameters for the pair and embedding function splines.

Hepburn FitPot

FitPot was created by Hepburn, a co-author of papers with Ackland[151]. It was written in Java and tackles the fitting as a non-linear least squares problem. The Levenberg-Marquardt Algorithm (LMA) is used to minimise the response function and uses the derivatives of the response function with respect to the input parameters to construct the Jacobian that is then used to construct an estimation of the Hessian.

Brommer PotFit

Originally developed in C, the PotFit[152] program uses the force matching method of fitting. In the early versions of the code, splines were the only function choice available. In the years since, the option to fit analytic functions has been added. Simulated annealing is the primary search algorithm that attempts to locate the

global minima. Powell's conjugate direction method is used to locate the local minima once the probable global minima is located. In order to use PotFit, a database of force, energy and stress for various configurations is required.

Sheng Modified PotFit

A set of potentials have been derived by Sheng et al[153] and are available to download in tabulated form[43]. They were created using a modified version of the PotFit code by P. Brommer and F. Gähler[152]. The code had been modified extensively to include quintic spline interpolation, phonon calculation, elastic constant calculation and optimization techniques[43]. The potentials (typically) used 15 knot splines for pair and density functions and 6 knot splines for the embedding functional for each element, but the potentials were published as tabulated functions. Potentials for fourteen FCC metals were derived in the first instance, but more (including alloys) are available on the website. Unfortunately, the (Sheng et al.) modified PotFit code is not available through the PotFit website[152], and the latest version of their code has moved on since 2011.

General Utility Lattice Program

General Utility Lattice Program (GULP)[154] is a Fortran 90 code that has been developed by Julian Gale since the late 90s. It is currently in its 6th version and is a large code at over 750,000 lines of code. It is able to compute many properties of a material and is able to minimise the energy of a system and fit potentials. Early in this work, GULP was considered as a possible tool to wrap a potential fitting code around, but without an API the first approach relied on writing input files and reading output files, which was inefficient.

The author notes in Chapter 1 of the GULP manual that it is a large code. Adding the desired forms of potential for radiation damage that included splines and a ZBL core pair potential would have required knowledge of the code and a lot of time to program. Adding new types of potential such as the 2BEAM would have resulted in the same issue.

5.7 Density Functional Theory

Density Functional Theory (DFT) is a branch of quantum chemistry that approximately solves the Schrödinger equation using electron density, rather than the coordinates of each electron in the system. A number of simplifications are also applied in order for DFT to be practical to use (fig. 5.12), but despite this calculations are limited to just hundreds or thousands of atoms. A calculation of a hundred or so atoms may take thousands of CPU hours at the time of writing, depending on the type of calculation and complexity of the electron structures of the atoms involved.

It is through DFT that the first principles energy, stress and force calculations will be made, and it is these results that the EAM potentials will be trained and fit to using the force matching method. This will allow much larger scale modelling using the extrapolated behaviour of accurate DFT calculations.

5.7.1 Motivation For Using DFT In This Work

The use of interatomic potentials bridges the gap between the quantum scale and macroscopic scale. In order to develop these potentials either measurements or calculated values are needed for forces, energies and stresses. DFT is a convenient choice and has been used to fit potentials and calculate crystal properties many times (Sheng et al[153], Mehl et al.[124] Connetable and Thomes,[155] Ravindran et al[130], Hepburn and Ackland[151] and more).

5.7.2 Brief Overview of DFT

Several important theories and approximations are used by DFT with the aim of calculating and minimising energies and forces. The Born-Oppenheimer approximation separates the electron-nucleus wave function. It treats the nuclei as fixed points, and the system of electrons in a fixed potential created by the nuclei.

The DFT of Kohn, Sham and Hohenberg proved that the potential of a system is uniquely determined by its ground state density. This makes solving the Schrödinger equation significantly easier.

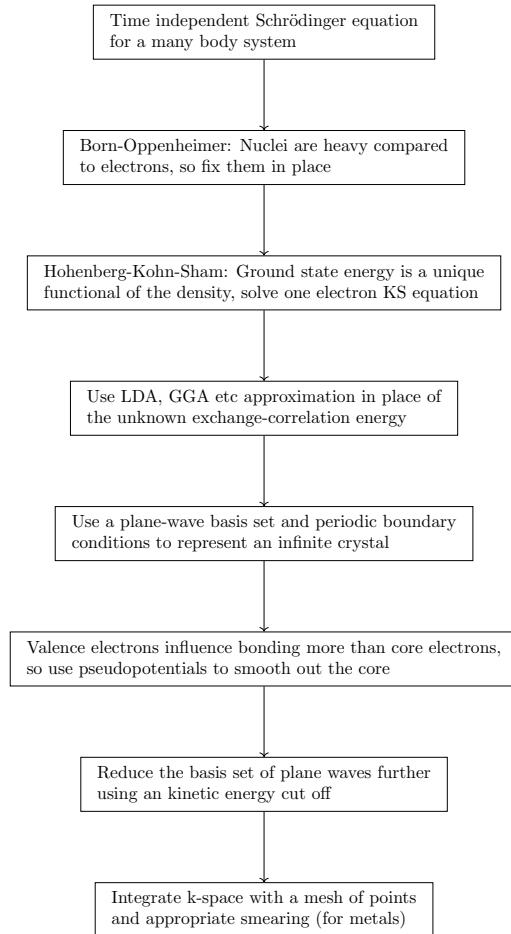


Figure 5.12: Common approximations used to enable DFT calculations

5.7.3 Time Independent Schrödinger Equation

The Schrödinger equation is a linear partial differential wave equation and it was proposed by Erwin Schrödinger in 1925. There is a time-dependent and time-independent form of the equation. As the DFT calculations in this work are static only the time-independent version will be discussed.

$$\hat{H}|\Psi\rangle = E|\Psi\rangle \quad (5.77)$$

The Hamiltonian \hat{H} is an operator and it is the total energy in the system. Ψ is the wave function and this contains all the measurable information possible about whatever it represents. E is the energy eigenvalue of this system and this will depend on the eigenstate of the system. The wave function is also connected to the probability of a particle being found at a certain point in space, and the integral over all space is equal to 1 i.e. the probability of it being found somewhere in space is equal to 1 (eq. 5.78).

$$\int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} |\Psi(x, y, z)|^2 dx dy dz = 1 \quad (5.78)$$

The Schrödinger equation is set up depending on the system being studied. Starting with the simplest, a free particle, the only non zero energy of the Hamiltonian is kinetic.

$$\begin{aligned} \hat{H} &= -\frac{\hbar^2}{2m} \nabla^2 \\ -\frac{\hbar^2}{2m} \nabla^2 \Psi(\vec{r}) &= E\Psi(\vec{r}) \end{aligned} \quad (5.79)$$

If the particle is in a potential, it will have both kinetic energy \hat{T} and potential energy \hat{V} (eq. 5.80).

$$\begin{aligned} \hat{H} &= \hat{T} + \hat{V} = -\frac{\hbar^2}{2m} \nabla^2 + V(\vec{r}) \\ \left[-\frac{\hbar^2}{2m} \nabla^2 + V(\vec{r}) \right] \Psi(\vec{r}) &= E\Psi(\vec{r}) \end{aligned} \quad (5.80)$$

5.7.4 Many Body TISE

Electronic structure calculations are important and allow the calculation of material properties that may be difficult or impossible to measure with current technology. The next step towards this is to set up the Schrödinger equation (time-independent) for nuclei and electrons in a crystal.

The only potential considered is that due to the electromagnetic force. The energy operators required are kinetic and electromagnetic potential (eq. 5.81).

$$\hat{H} = \hat{T}_e + \hat{T}_n + \hat{V}_{e-e} + \hat{V}_{e-n} + \hat{V}_{n-n} \quad (5.81)$$

The first two terms are the kinetic energy of the electrons and nuclei respectively (eq. 5.82).

$$\begin{aligned} \hat{T}_e &= -\sum_i \frac{\hbar^2}{2m} \nabla^2 \text{ sum of kinetic energy of electrons} \\ \hat{T}_n &= -\sum_k \frac{\hbar^2}{2M} \nabla^2 \text{ sum of kinetic energy of nuclei} \end{aligned} \quad (5.82)$$

The last three terms are potential energy terms due to the electromagnetic force (eq. 5.83).

$$\begin{aligned} V_{e-e} &= \frac{1}{2} \sum_{i,j, i \neq j} \frac{1}{|\vec{r}_i - \vec{r}_j|} \text{ sum of potential energy between electrons} \\ V_{e-n} &= \sum_{i,k} \frac{z_i}{|\vec{r}_i - \vec{r}_l|} \text{ sum of potential energy between electrons and nuclei} \\ V_{n-n} &= \frac{1}{2} \sum_{k,l, k \neq l} \frac{z_k z_l}{|\vec{r}_l - \vec{r}_k|} \text{ sum of potential energy between nuclei} \end{aligned} \quad (5.83)$$

These operators are now input into the TISE (eq. 5.84).

$$\left[\left(-\sum_i \frac{\hbar^2}{2m} - \sum_k \frac{\hbar^2}{2M} \right) \nabla^2 + \sum_{i,k} \frac{z_i}{|\vec{r}_i - \vec{r}_l|} + \frac{1}{2} \left(\sum_{i,j, i \neq j} \frac{1}{|\vec{r}_i - \vec{r}_j|} + \sum_{k,l, k \neq l} \frac{z_k z_l}{|\vec{r}_l - \vec{r}_k|} \right) \right] |\Psi\rangle = E|\Psi\rangle \quad (5.84)$$

5.7.5 Born-Oppenheimer

The TISE arrived at is far too complicated to solve, even for the simple ensembles of atoms. It represents the many body system of electrons and nuclei, and it takes the positions of all nuclei and electrons as input variables (eq. 5.85).

$$\begin{aligned} \hat{H}|\Psi(\vec{r}_e, \vec{r}_n)\rangle &= E|\Psi(\vec{r}_e, \vec{r}_n)\rangle \\ \text{where } \vec{r}_e &= r_{e,1}, r_{e,2}, \dots, r_{e,i} \text{ (electron positions)} \\ \text{where } \vec{r}_n &= r_{n,1}, r_{n,2}, \dots, r_{n,i} \text{ (electron positions)} \end{aligned} \quad (5.85)$$

In 1927 the Born-Oppenheimer approximation was proposed to separate the electron components from the nuclei in the Hamiltonian. Protons and neutrons are almost 2,000 times the mass of electrons. With respect to the electrons, they move much slower and may be considered to be fixed or frozen in place. Simplifying for a moment to a single electron and proton, due to Newton's second law, we can see that the acceleration of the electron would be similarly 2,000 times that of the proton: $f_e = f_p$ and $m_e a_e = m_p a_p$ which leads to $a_e = \frac{m_p}{m_e} a_p$. As the nuclei move, the electrons are assumed to respond instantly, remaining in the ground state and not being promoted into higher energy levels.

The Hamiltonian for the electrons may be written with the electron co-ordinates as a variable, and the nuclei co-ordinates as a parameter (eq. 5.86).

$$\hat{H}_e(\vec{r}_e; \vec{r}_n) = \hat{T}_e(\vec{r}_e) + \hat{V}_{e-e}(\vec{r}_e) + \hat{V}_e - \hat{n}(\vec{r}_e; \vec{r}_n) \quad (5.86)$$

The wavefunction and energy for the electrons may be calculated, although if the nuclear coordinates are changed, i.e. by changing the parameter r_n , the wavefunction and energy will need to be recalculated.

$$\hat{H}_e(\vec{r}_e; \vec{r}_n) \psi_e(\vec{r}_e; \vec{r}_n) = E_e(\vec{r}_n) \psi_e(\vec{r}_e; \vec{r}_n) \quad (5.87)$$

$$\Psi(\vec{r}_e, \vec{r}_n) = \chi_{ne}(\vec{r}_e) \psi_e(\vec{r}_e; \vec{r}_n) \quad (5.88)$$

The electronic Hamiltonian (eq. 5.87) is still dependent on the electronic co-ordinates. As there are three co-ordinates per electron, there are 3 dimensions when solving for a hydrogen atom. For an Iron atom, with 26 electrons, there are 78 dimensions, and for a 4x4x4 FCC supercell of Iron there would be 256 atoms, 26 electrons per atom and 3 dimensions per electron, giving a total of 19,968 dimensions. Even for small numbers of electrons, solving this equation is impractical.

5.7.6 Crystals, Reciprocal Space and Bloch Theorem

A Bravais lattice is a construct used to describe a periodic crystal lattice. It has the following properties given in eq. 5.89.

$$\vec{R} = n_1 \vec{a}_1 + n_2 \vec{a}_2 + n_3 \vec{a}_3$$

$$n_1, n_2, n_3 \in \mathbb{Z}$$

$$\vec{a}_1, \vec{a}_2, \vec{a}_3 \text{ are independent}$$
(5.89)

There are 14 Bravais lattices and 7 families of these lattices, and these lattices may be grouped by vector length and angle relationships (table. 5.4). This work is primarily concerned with cubic and tetragonal crystals, although distortions are applied to these crystals throughout.

Class	Lengths	Angles
Cubic	$a = b = c$	$\alpha = \beta = \gamma = 90$
Hexagonal	$a = b, c$	$\alpha = \beta, \gamma = 120$
Rhombohedral	$a = b = c$	$\alpha = \beta = \gamma \neq 90$
Tetragonal	$a = b, c$	$\alpha = \beta = \gamma = 90$
Orthorhombic	a, b, c	$\alpha = \beta = \gamma = 90$
Monoclinic	a, b, c	$\alpha = \beta = 90, \gamma \neq 90$
Triclinic	a, b, c	$\alpha, \beta, \gamma,$

Table 5.4: Bravais lattice vector length and angle relationships

Bloch Theorem

Metals are made up from grains which in turn are crystal lattices of atoms. Despite the majority of metals being composed of a large collection of microscopic crystals, rather than being a single perfect crystal, most of their properties may be calculated as if the metal were a single crystal.

The grain size of water quenched SS304 is approximately 30 micrometers across[156]. If the crystal were a cube, it would contain 2.0×10^{15} atoms and it would have sides over 100,000 atoms in length.

In solid state physics, in order to solve the TISE for such a system, it is useful to consider an infinite sized crystal. It may be helpful to visualise this as a “ring” of atoms in one dimension (fig. 5.13) or as a super-cell with periodic boundary conditions in three dimensions.

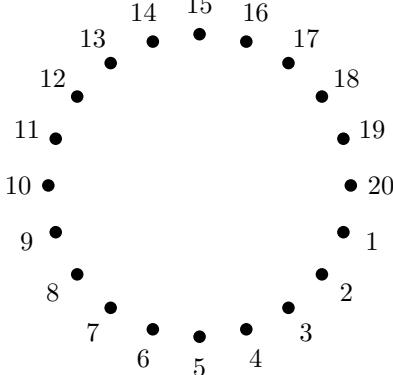


Figure 5.13: A useful, albeit incorrect, way of visualising an "infinite" chain of atoms in 1D

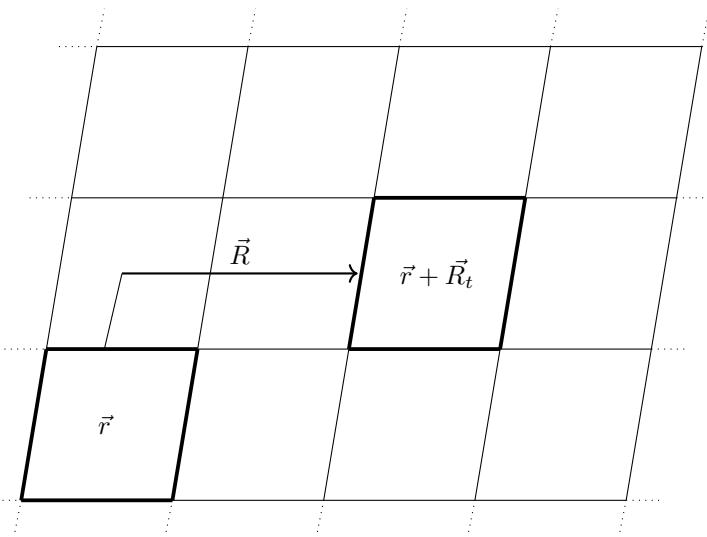


Figure 5.14: A 2D example of a translation by an integer multiple of the unit cell from the origin unit cell

As the structure is a repeating lattice, any point within a unit cell is equivalent to any other point translated by an integer multiple of the lattice vector (fig. 5.14).

$$\psi(\vec{r}) = \psi(\vec{r} + \vec{R}_t) \quad (5.90)$$

where $\vec{R}_t = n_1 \vec{R}_1 + n_2 \vec{R}_2 + n_3 \vec{R}_3$

where \vec{R} is the real lattice vector

Reciprocal space, also known as k-space or momentum space, is a mathematical construct. It is an imaginary space where the lengths, and volumes, are the inverse of real space and planes of atoms are represented as points.

Starting with a lattice of points in real space \vec{R} , points in reciprocal space \vec{G} only are valid points if $\exp(i\vec{G} \cdot \vec{R}) = 1$ [157]. A real space vector \vec{R} is transformed to its reciprocal in the following way, where Ω is the volume of the primitive cell in real space.

$$\begin{aligned} \vec{g}_1 &= \frac{2\pi}{\Omega} \vec{r}_2 \times \vec{r}_3 \\ \vec{g}_2 &= \frac{2\pi}{\Omega} \vec{r}_3 \times \vec{r}_1 \\ \vec{g}_3 &= \frac{2\pi}{\Omega} \vec{r}_1 \times \vec{r}_2 \\ \Omega &= \vec{r}_1 \cdot (\vec{r}_2 \times \vec{r}_3) \end{aligned} \quad (5.91)$$

If the crystal structure is modelled as an infinitely repeating lattice, the potential also has the same periodicity (eq. 5.91).

$$V(\vec{r}) = V(\vec{r} + \vec{R}_t) \quad (5.92)$$

As the potential is a periodic function, it may be written as a Fourier series in reciprocal space (eq. 5.93).

$$V(\vec{r}) = \sum_{\vec{G}} V_{\vec{G}} \exp(i\vec{G} \cdot \vec{r}) \quad (5.93)$$

The wave function for an electron ($\psi_{n\vec{k}}$) in the lattice is assumed periodic with the lattice structure (eq. 5.94) and may be written as the product of a plane wave and a cell periodic function ($u_{n\vec{k}}$), where n denotes the band and \vec{k} is the wave vector.

$$u_{n\vec{k}}(\vec{r}) = u_{n\vec{k}}(\vec{r} + \vec{R}_t) \quad (5.94)$$

$$\begin{aligned} \psi_{n\vec{k}}(\vec{r}) &= u_{n\vec{k}}(\vec{r}) \exp(i\vec{k} \cdot \vec{r}) \\ \psi_{n\vec{k}}(\vec{r}) &= \psi_{n\vec{k}}(\vec{r}) \exp(i\vec{k} \cdot \vec{R}_t) \end{aligned} \quad (5.95)$$

The Born-von Karman boundary conditions apply eq. 5.94 and eq. 5.95 to the wave function such that:

$$\psi_{n\vec{k}}(\vec{r} + \vec{R}_t) = \psi_{n\vec{k}}(\vec{r}) \text{ where } \vec{R}_t = \sum_i N_i \vec{a}_i \text{ and } N_i \in \mathbb{Z} \quad (5.96)$$

With these boundary conditions and use of plane waves, Bloch theorem replaces an enormous number of electrons with an infinite number electrons in a periodic lattice where only those in the unit cell need to be considered.

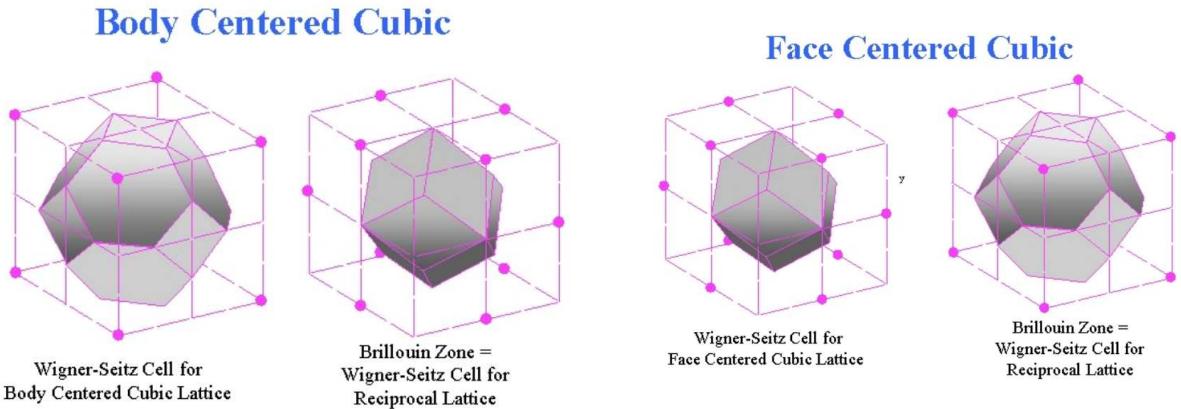


Figure 5.15: BCC Wigner Seitz Cell and BZ [158]

Figure 5.16: FCC Wigner Seitz Cell and BZ [158]

The (first) Brillouin Zone is a volume in (3D) reciprocal space and it is analogous to the Wigner-Seitz cell in real space (figs. 5.15 and 5.16). It is the primitive lattice cell in reciprocal space. The reciprocal of a BCC lattice is a FCC lattice (fig. 5.15) and vice versa (fig. 5.16). In real space a point anywhere in the lattice is a integer multiple of the lattice vector away from the same point in the origin Wigner-Seitz cell. There are an infinite number of wave vectors \vec{k} in reciprocal space, but by virtue of the periodicity of the lattice, they are also all found within the Brillouin Zone. If the crystal has symmetries, the BZ may be reduced further to the irreducible Brillouin zone.

Within the BZ there are energy surfaces. Picking a straight line through reciprocal space allows one to plot the energy bands along that one dimensional space. A particularly important energy surface is the Fermi surface

and, in the one electron model, this marks the boundary between the occupied and unoccupied states at 0K [159]. The Fermi energy is the energy of the highest occupied state at 0K.

5.7.7 Hohenberg-Kohn Theorem

In the 1960s, Hohenberg and Kohn[160] simplified the problem of solving the many electron TISE in an exact way by proving:

- in an external potential $v(\vec{r})$, the potential is uniquely determined by the density of the ground state $n_0(\vec{r})$ assuming that the particles are non-degenerate
- a uniquely defined functional $E[\rho(\vec{r})]$ exists and the ground state energy, $\min(E[\rho(\vec{r})])$, is found by varying the density

The proof starts by picturing a box of electrons that interact with each other through coulomb repulsion within an external potential $\vec{v}(r)$, for example the potential of “fixed” nuclei following the Born Oppenheimer approximation.

$$\begin{aligned} \hat{H} &= \hat{T} + \hat{V} = -\frac{1}{2}\nabla^2 + V(\vec{r}) \\ \left[-\frac{1}{2}\nabla^2 + V(\vec{r})\right]\Psi(\vec{r})_0 &= E_0\Psi(\vec{r}) \end{aligned} \quad (5.97)$$

$$\hat{H} = \hat{T} + \hat{V} + \hat{U} \quad (5.98)$$

where

$$T = \frac{1}{2} \int \nabla\psi^*(\vec{r})\nabla\psi(\vec{r})d\vec{r} \quad (5.99)$$

$$V = \int v(\vec{r})\psi^*(\vec{r})\psi(\vec{r})d\vec{r} \quad (5.100)$$

$$U = \frac{1}{|\vec{r} - \vec{r}'|}\psi^*(\vec{r})\psi^*(\vec{r}')\psi(\vec{r})\psi(\vec{r}')d\vec{r}d\vec{r}' \quad (5.101)$$

In the eqs. 5.98 5.99 5.98 5.101 the hamiltonian operator \hat{H} is a sum of the kinetic energy \hat{T} , the mutual coulomb repulsion \hat{U} and the operator due to an external potential (due to the nuclei) \hat{V} . The non degenerate ground state electron density is $n(\vec{r}) = \langle \Phi | \phi^*(\vec{r})\phi(\vec{r}) | \Phi \rangle$.

The HK theorem shows that the external potential $v(\vec{r})$ is a unique functional of the electron density $n(\vec{r})$. The proof is that by contradiction where by two different states are assumed to have the same ground state charge density.

State A

$$\begin{aligned}\Psi_A \text{ with potential } V_A(\vec{r}) \\ \text{Hamiltonian } \hat{H} = \hat{T} + \hat{V}_A + \hat{U} \\ \text{TISE } \hat{H}_A \Psi_A = E_A \Psi_A\end{aligned}$$

Assumption - this state has the charge density $n(\vec{r})$

The minimal property of the ground state gives the following[160][161]:

State B

$$\begin{aligned}\Psi_B \text{ with potential } V_B(\vec{r}) \\ \text{Hamiltonian } \hat{H} = \hat{T} + \hat{V}_B + \hat{U} \\ \text{TISE } \hat{H}_B \Psi_B = E_B \Psi_B\end{aligned}$$

Assumption - this state has the charge density $n(\vec{r})$

$$\begin{aligned}E_A &= \langle \Phi_A | \hat{H}_A | \Phi_A \rangle \\ E_A &= \langle \Phi_A | \hat{H}_B + \hat{V}_A - \hat{V}_B | \Phi_A \rangle \\ E_A &= \langle \Phi_A | \hat{H}_B | \Phi_A \rangle + \int d^3 \vec{r} n(\vec{r}) (v_A(\vec{r}) - v_B(\vec{r}))\end{aligned}\tag{5.102}$$

$$E_A > E_B + \int d^3 \vec{r} n(\vec{r}) (v_A(\vec{r}) - v_B(\vec{r}))\tag{5.103}$$

The indices A and B are interchanged to give a second equation:

$$\begin{aligned}E_B &> E_A + \int d^3 \vec{r} n(\vec{r}) (v_B(\vec{r}) - v_A(\vec{r})) \\ E_B &> E_A - \int d^3 \vec{r} n(\vec{r}) (v_A(\vec{r}) + v_B(\vec{r}))\end{aligned}\tag{5.104}$$

By adding eq. 5.103 and eq. 5.104 together the integrals will cancel out.

$$E_A + E_B > E_A + E_B\tag{5.105}$$

This (eq. 5.105) is a contradiction and proves the first part of the HK theorem: $v(\vec{r})$ is uniquely determined by $n(\vec{r})$. If the charge density is known, a single external potential exists for it.

The next part of the theorem is to show that the energy functional $E[n(\vec{r})]$ exists and the minimum can be found by varying the charge density function. The kinetic and interaction functional \hat{T} and \hat{U} define a functional $F[n(\vec{r})]$.

$$F[n(\vec{r})] \equiv \langle \Phi | \hat{T} + \hat{U} | \Phi \rangle\tag{5.106}$$

This is a universal functional[160] and it makes up a part of the energy functional. The energy functional is comprised of the HK functional $F[n(\vec{r})]$ and the external potential as shown in eq. 5.107.

$$E[n(\vec{r})] \equiv \int v(\vec{r}) n(\vec{r}) d\vec{r} + F[n]\tag{5.107}$$

For the ground state electron density $n(\vec{r})$ the energy will be equal to the ground state energy $E_0 = E_v[n]$. Where N is the number of particles in the system, the integral of the density over all space will sum to N (eq. 5.108).

$$N[n] = \int n(\vec{r}) d\vec{r} \quad (5.108)$$

If the system has N particles the energy level for a none ground state energy, k, is given in eq. 5.109.

$$\epsilon_{v,k}[\Psi_k] \equiv \langle \Phi_k | \hat{V} | \Phi_k \rangle - \langle \Phi_k | \hat{T} - \hat{U} | \Phi_k \rangle \quad (5.109)$$

This has a minimum where $\Psi_k = \Psi_0$, the ground state.

$$\begin{aligned} \epsilon_{v,k}[\Psi_k] &= \int v(\vec{r}) n_k(\vec{r}) d\vec{r} + F[n_k] \\ \epsilon_{v,0}[\Psi_k] &= \int v(\vec{r}) n_0(\vec{r}) d\vec{r} + F[n_0] \\ \epsilon_{v,k} &> \epsilon_{v,0} \end{aligned} \quad (5.110)$$

5.7.8 Kohn-Sham Equations

The year following the Hohenberg-Kohn theorem, a set of self-consistent equations were derived by Kohn and Sham. The ground state energy of interacting jellium in the potential of fixed nuclei, an external potential, can be written as follows[162]:

$$E = T_s + U + V_{n-e} + E_{xc} \quad (5.111)$$

$$E = T_s[\rho(\vec{r})] + \int d\vec{r} v(\vec{r}) \rho(\vec{r}) + \frac{1}{2} \int \int d\vec{r} d\vec{r}' \frac{\rho(\vec{r}) \rho(\vec{r}')}{|\vec{r} - \vec{r}'|} + E_{xc}[\rho(\vec{r})] \quad (5.112)$$

The kinetic energy, T_s , is that of a system of non interacting particles and E_{xc} is the exchange and correlation of an interacting system. The exchange and correlation energy functional exists, but is unknown.

The Kohn-Sham equations are used to calculate the energy of a many body Schrödinger equation by solving for a single electron.

$$\hat{H}_{KS} \psi_i = E_i \psi_i \quad (5.113)$$

$$\left(-\frac{1}{2} \nabla^2 + \hat{v}_{KS}[n](\vec{r}) \right) \psi_i(\vec{r}) = E_i \psi_i(\vec{r}) \quad (5.114)$$

$$\hat{v}_{KS}(\vec{r}_e, \vec{r}_n) = v_{n-e}(\vec{r}_e, \vec{r}_n) + \int d^3 \vec{r}' \frac{\rho(\vec{r}'_e)}{|\vec{r}_e - \vec{r}'_e|} + v_{xc}[\rho](\vec{r}_e) \quad (5.115)$$

$$n(\vec{r}) = \sum_i^N |\psi_i(\vec{r})|^2 \quad (5.116)$$

It is important to note that even though this is a one electron equation it is an exact solution.

5.7.9 Self-Consistent Solution

The Kohn-Sham equations cannot be solved in the usual way. They require a value for the density, but the density is obtained by solving the equations, so there is a dilemma. It can however be solved self consistently: an initial density is guessed, and this is repeatedly updated until the density in and out values are the same (or within a set convergence threshold). The basic algorithm used by PWscf from the Quantum Espresso suite is shown in fig. 5.17[163].

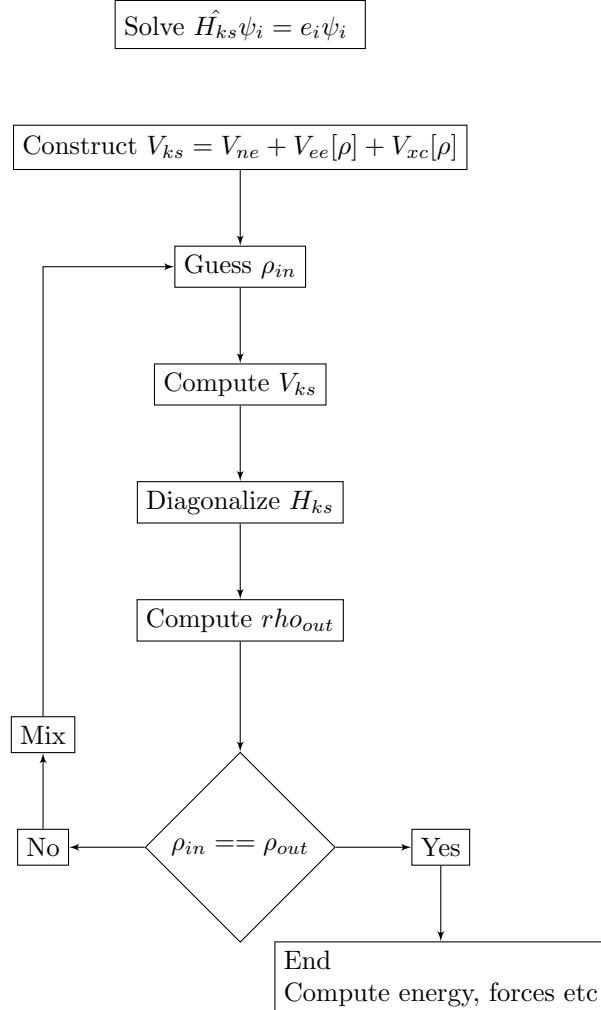


Figure 5.17: SCF algorithm for Quantum Espresso

In practice the input and output densities may never be exactly the same so rather than wait indefinitely for convergence, a threshold is set. The smaller the threshold the more iterations and time would be expected to pass, but the calculation will be closer to the solution.

With each iteration the density of the last iteration is mixed into a new density. If simple linear mixing is used a mixing beta determines the ration of new to old density as given in eq. 5.117[164].

$$\rho^{n+1} = (1 - \beta)\rho_{KS}^n + \beta\rho_{out}^n \quad (5.117)$$

More advanced mixing methods include Newton-Raphson or Broyden mixing. The PWscf program within Quantum Espresso uses Broyden for charge density mixing by default. Charge sloshing is an issue where convergence is slow or never happens. This is a particular problem for metals[165] and is more significant the larger the system[166]. The Thomas-Fermi screening used is based on the work by Johnson[167] and this

works well for highly homogenous systems. Further work by Raczkowski, Canning and Wang[166] developed a Pulay-Thomas-Fermi method that is implemented in PWscf with the local-TF mixing mode, and this is better suited to highly inhomogeneous systems.

5.7.10 Exchange-Correlation Energy

The Kohn-Sham equations have an exchange-correlation energy and this functional is used to collect together the electron energy not captured in the non-interacting energy functionals.

The Pauli exclusion principle states that two fermions, half integer spin particles, cannot occupy the same quantum state. This is why electrons occupy a unique orbital within an atom defined by the quantum numbers n, l, m_l and m_s . Consider two particles at points \vec{r}_a and \vec{r}_b ; the probability amplitude of the wavefunction of these particles equals 1.

$$|\Psi(\vec{r}_a, \vec{r}_b)|^2 = 1 \quad (5.118)$$

Exchanging the particles must also give the same result; they still must exist somewhere with probability 1.

$$|\Psi(\vec{r}_b, \vec{r}_a)|^2 = 1 \quad (5.119)$$

Bosons, integer spin particles, are symmetric when exchanged:

$$\Psi(\vec{r}_a, \vec{r}_b) = \Psi(\vec{r}_b, \vec{r}_a) \quad (5.120)$$

Fermions, on the other hand, are antisymmetric:

$$\Psi(\vec{r}_a, \vec{r}_b) = -\Psi(\vec{r}_b, \vec{r}_a) \quad (5.121)$$

By setting up a wavefunction for two Fermions, and exchanging them, it can be seen why they cannot exist in the same state.

$$\Psi_{ab} = \psi_1(\vec{r}_a, \vec{r}_b) - \psi_2(\vec{r}_b, \vec{r}_a) \quad (5.122)$$

$$\Psi_{ab} = \psi_1(\vec{r}_a, \vec{r}_b) - \psi_2(\vec{r}_a, \vec{r}_b) = 0 \quad (5.123)$$

The XC term combines the difference between the real system and the fictitious non-interacting system set out in the Kohn-Sham equations. It also includes the difference between the quantum mechanical electron-electron repulsion and classical electron-electron repulsion. Unfortunately, whilst we know a functional exists, we do not know the exact form of it[168].

5.7.11 Pseudopotentials

Plane wave basis sets are used to help solve the TISE. The DFT code used in this work is Quantum Espresso, and the binary that carries out the calculations is named PWscf: plane wave self consistent field.

Where \vec{G} is the reciprocal lattice vector, a summation of plane waves may be used to construct each electronic wave function (eq. 5.124)[169].

$$\Psi_{\vec{k},n} = \sum_{|\vec{G}| < G_{max}} c_{\vec{k}+\vec{G},n} \exp(i(\vec{k} + \vec{G})\vec{r}) \quad (5.124)$$

Unfortunately, the plane-wave basis sets required are too large to be used in practice, as they would need to represent the tight, rapidly oscillating inner orbitals as well as the valence electrons.

Bonding and material properties are primarily determined by valence electrons, not core electrons. Iron, for example, has two valence electrons in the 4s shell; however, it is a transition metal and the partially empty 3d shell is also important to consider. The core electrons do not contribute as much to the bond and the model may be simplified using pseudopotentials.

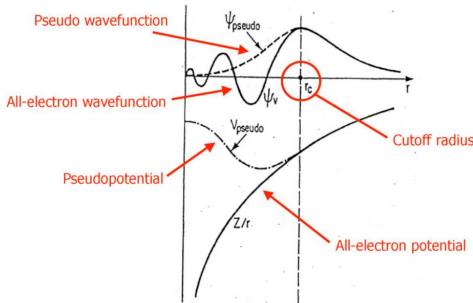


Figure 5.18: Replacing the complex potentials with a pseudopotential[170]

The eigen-energies of core electrons are also much larger than properties we see in materials, such as the cohesive energies and this also raises a concern that including the core electrons may introduce errors that are on a similar scale to the energies we are calculating.

The rapidly oscillating function in the core is replaced by a smoother pseudopotential (fig. 5.18). The same material properties are calculated as the valence electron functions are preferred, but a much smaller basis set is required to do so. There are a wide range of pseudopotentials available to use in DFT calculations, and one major distinguishing feature is how they treat the XC potential $v_{xc}([n]; \vec{r})$.

$$\left(\frac{1}{2} + V_{ext}(\vec{r}) + V_H(\vec{r}) + V_{xc}(\vec{r}) \right) \psi_{k,\sigma}(\vec{r}) = \epsilon_{k,\sigma} \psi_{k,\sigma}(\vec{r}) \quad (5.125)$$

The KS equation consists of several potentials including the exchange-correlation (XC) energy. This term represents the many-electron effects.

LDA

The local density approximation replaces the electron density with jellium, a homogeneous electron gas. Kohn and Sham, in their original 1965 work, introduced jellium as the density model and, despite its simplicity in comparison to the real electron density distribution, it has worked well for simple metals.

The XC energy for the LDA is an integral over space in the local neighbourhood of the point of interest. Both the exchange term (e_x) and correlation term (e_c) are functionals of the local electron density ($\rho(\vec{r})$).

$$E_{xc}^{LDA}[\rho] = \int d^3\vec{r}\rho(\vec{r})[e_x(\rho(\vec{r})) + e_c(\rho(\vec{r}))] \quad (5.126)$$

Several different LDA functionals have been developed over the years, including that of Perdew and Zunger. This functional takes the form given in eq. 5.127[171].

$$\epsilon_C[n(\vec{r})] = \begin{cases} \frac{-0.14231}{1+1.9529r_s^{(0.5)}+0.334r_s} & r_s \geq 1 \\ -0.0480 + 0.0311 \ln(r_s) - 0.0116r_s + 0.0020r_s \ln(r_s) & r_s < 1 \end{cases} \quad (5.127)$$

LSDA

Electrons are fermions and have half integer spin. When trapped in a potential, such as an atom or a crystal lattice, electrons must take difference quantum states and one of the parameters is the spin of the electron. The LDA does not treat spin at all, but the LSDA splits the density into spin up (ρ_\uparrow) and down spin (ρ_\downarrow) (eq. 5.128[168]).

$$\begin{aligned} \rho_\uparrow(\vec{r}) &= \sum_k^{occ} |\phi_{k,\uparrow}(\vec{r})|^2 \\ \rho_\downarrow(\vec{r}) &= \sum_k^{occ} |\phi_{k,\downarrow}(\vec{r})|^2 \\ \rho(\vec{r}) &= \rho_\uparrow(\vec{r}) + \rho_\downarrow(\vec{r}) \end{aligned} \quad (5.128)$$

As with the LDA the exchange and correlation terms are separate but the XC energy is now dependent on the spin up and spin down density as well as the position (eq. 5.129[168]), where $\zeta(\vec{r})$ is the relative spin-polarisation. Setting this to 0 reverts this to the spin unpolarised case.

$$\begin{aligned} E_{xc}^{LSDA}[\rho_\uparrow, \rho_\downarrow] &= \int d^3\vec{r}\rho(\vec{r})[e_x(\rho(\vec{r}))f(\zeta(\vec{r})) + e_c(\rho(\vec{r}), \zeta(\vec{r}))] \\ \zeta &= \frac{\rho_\uparrow - \rho_\downarrow}{\rho_\uparrow + \rho_\downarrow} \\ f(\zeta) &= \frac{1}{2} \left((1 + \zeta)^{4/3} + (1 - \zeta)^{4/3} \right) \end{aligned} \quad (5.129)$$

The energy of the system now being solved is also dependent upon the charge density and spin polarization (eq. 5.130).

$$E = T[\rho, \zeta] + E_{ext}[\rho] + \frac{1}{2}E_{ee}[\rho] + E_{xc}[\rho, \zeta] \quad (5.130)$$

Many correlation energy functionals have been developed in order to improved the LSDA and it is still widely used as will be discussed in section 5.7.11. There are now more complicated approaches including the GGA.

GGA

In the LDA and LSDA approximations, the electron density is that of an homogeneous electron gas that does not represent the real electron density. The density of this gas may change in space, but locally it is a constant density with no gradient.

$$E_{xc}^{GGA}[\rho_\uparrow, \rho_\downarrow] = \int d^3r f(\rho_\uparrow, \rho_\downarrow, \nabla \rho_\uparrow, \nabla \rho_\downarrow) \quad (5.131)$$

The GGA functional (eq. 5.131 [168]) takes the spin up (ρ_\uparrow) spin down(ρ_\downarrow) electron densities as well as the gradients of the spin densities ($\nabla \rho_\uparrow, \nabla \rho_\downarrow$).

The PBE functional was developed to address some of the short falls of the Perdew-Wang PW91 LSDA functional. These include an over complication in the formulation of PW91 and that it was designed to satisfy many exact conditions. The PBE development focused more on energetically significant conditions[172].

$$E_c^{PBE}[\rho_\uparrow, \rho_\downarrow] = \int d^3r \rho [e_c(r_s, \zeta) + H(r_s, \zeta, t)] \quad (5.132)$$

$$E_x^{PBE}[\rho] = \int d^3r \rho e_x(\rho) F_x(s) \quad (5.133)$$

The functional is split into a correlation energy (eq. 5.132) and exchange energy (eq. 5.133). ζ is the relative spin polarization $\zeta = (\rho_\uparrow - \rho_\downarrow)/(\rho_\uparrow + \rho_\downarrow)$ and r_s is the local Seitz radius where $\rho = 3/4\pi r_s^3$. The functions $e_x(\rho)$ and $e_c(\rho)$ are the exchange and correlation energy per electron of unpolarised uniform electron gas[168].

Both functionals $H(r_s, \zeta, t)$ and $F_x(s)$ are dependent upon t and s and these in turn are dependent upon the density and density gradients. Derived by Perdew, Burke and Ernzerhof[172], there is a full description outlined in a later review by Ziesche, Kurth and Perdew[168]. A full description of the functional and the terms is also included in appendix J.

LSDA vs GGA Potentials

Typically, GGA is an improvement over LSDA, with improvements in the total energies and structural energy differences[172]. Another short coming of LSDA is that it predicts FCC to be the optimal structure for pure iron at 0K, which is incorrect[173]. A list of success stories and failures were discussed shortly after the publication of the PBE functional and these are summarised in fig. 5.19[168].

In the literature, investigations into comparisons between GGA, LDA and experiments have been carried out for Titanium Disilicide ($TiSi_2$) and Lanthanum aluminate ($LaAlO_3$). This work is briefly discussed to highlight the fact that certain pseudopotentials and exchange functionals better reproduce a particular material and property.

In the case of the compound $TiSi_2$, GGA is in better agreement, overall, with experiment than LDA, although several values are better predicted by the latter. As shown in table 5.5, the data shows that LDA is within 1.26% of the experimental lattice parameters, 6.16% of the experimental bulk modulus and 18.3% the value of the experimental elastic constants; several of these were particularly poor, with an almost 45% disagreement. On the other hand, GGA is within 0.66% of the experimental lattice parameters, 3.62% of the experimental bulk modulus and 14.97% the value of the experimental elastic constants[130].

Successes of GGA

- atomization energy of molecules better
- binding energy curves more realistic
- Fe is bcc ferromagnetic with GGA (fcc non-magnetic with LSDA)
- Gives the anti invar effect in gamma-Fe and fc Fe-Mn
- Improvement on 4% accuracy for LSDA of alkali metal lattice constants
- Better lattice constants and bulk moduli of transition metals
- Isostructural transformation from open to close packed improved
- Successful calculation of a monovacancy in silicon
- Oxidation of the Si(001) surface using spin polarized GGA

Failures

- exchange-correlation holes can be unrealistic under some circumstances
- works better for exchange correlation together than either alone
- interaction of electrons in different shells poorly described

Figure 5.19: Successes and failures of the GGA functional[168]

LaAlO_3 has also been studied using DFT, but this material is better modelled (to reproduce lattice parameters and bulk modulus) with either LDA or a specific form of GGA, the Perdew-Burke-Ernzerhof functional revised for solids (PBESOL) potential. The PBESOL is a revised PBE functional designed to better reproduce lattice parameters for solids, as shown in table 5.6.

A complete library of pseudo-potentials is available through the Quantum Espresso website. Several of these have been tested with the PWscf code to calculate the lattice parameter and bulk modulus of simple metals including Aluminium and Sodium.

As can be seen from table 5.7, the PBE and PBESOL GGA type potentials are in general better at reproducing the lattice parameters. However, for Aluminium, the PZ LSDA type potential is in better agreement when calculating the bulk modulus. Overall, the GGA type potentials with the total self-consistent potential pseudized (these are the pseudopotential files that contain -n- in the name, rather than -nl-) perform best for these simple metals.

Whilst DFT is an exact method for solving the TISE, there are a number of approximations that still need to be made in addition to there being gaps in our knowledge (i.e. a lack of an exact functional for the XC).

5.7.12 Ecut, K-Point Integration and Smearing

As discussed in section 5.7.6 the wavefunction can be expressed as a periodic function with the same period as the crystal lattice (eq. 5.134). The periodic function may then be written as a sum of plane waves (eq. 5.135).

$$\psi_{n\vec{k}}(\vec{r}) = u_{n\vec{k}}(\vec{r}) \exp(i\vec{k} \cdot \vec{r}) \quad (5.134)$$

$$u_{n\vec{k}}(\vec{r}) = \sum_{\vec{G}} V_{\vec{G}} \exp(i\vec{G} \cdot \vec{r}) \quad (5.135)$$

Parameter	Experimental	LDA	GGA
a/angs	8.27	8.08	8.21
b/angs	4.80	4.74	4.81
c/angs	8.55	8.53	8.64
B_0 (Reuss)/GPa	146.8	156.9	141.9
B_0 (Voigt)/GPa	150.9	159.1	145.0
C_{11} /GPa	317.5	377.2	326.0
C_{22} /GPa	320.4	341.1	298.4
C_{33} /GPa	413.2	425.3	371.9
C_{44} /GPa	112.5	136.5	123.5
C_{55} /GPa	75.8	93.7	85.3
C_{66} /GPa	117.5	154.6	135.9
C_{12} /GPa	29.3	27.8	22.4
C_{13} /GPa	38.4	21.3	26.5
C_{23} /GPa	86.0	95.1	105.5

Table 5.5: Experimental vs LSDA vs GGA for $TiSi_2$ [130]

Parameter	Experimental	LDA	GGA	GGA (PBESOL)
a/angs	3.78	3.74	3.82	3.77
B_0 /GPa	215	224.00	190.35	206.41

Table 5.6: Experimental vs LSDA vs GGA vs GGA PBESOL for $LaAlO_3$ [174]

Solving the Kohn-Sham equations requires the matrix diagonalization of a size NxN where N is the number of planewaves for each k-point in the Brillouin Zone[178]. The number of planewaves can be reduced to satisfy eq. 5.136. In practice this will mean reducing E_{cut} whilst ensuring the desired accuracy is still met.

$$\frac{\hbar^2 G^2}{2m} \leq E_{cut} \quad (5.136)$$

In order to compute the energy using HK, the charge density in the BZ is calculated. It is impossible to diagonalise the matrix at an infinite number of k-points. The integral is replaced with a summation over selected points which in turn are weighted (eq. 5.137)[179].

$$\rho(\vec{r}) = \frac{1}{\Omega_{bz}} \sum_i \int f(\vec{k}, i) \psi_{i,\vec{k}}^*(\vec{r}) \psi_{i,\vec{k}}(\vec{r}) d\vec{k} \frac{1}{\Omega_{bz}} \int_{BZ} d\vec{k} \rightarrow \sum_{\vec{k}} \omega_{\vec{k}} \quad (5.137)$$

Due to Bloch theorem and the Born-Von-Karman boundary conditions, only the first BZ needs to be sampled. As discussed earlier, if there are further symmetries in the unit cell, only the IBZ need to be sampled.

The integral over k-space is replaced by a set of points in k-space that are sampled. The higher the number of points, the higher resolution the space is sampled in, but the longer the calculation will take. The Gamma point Γ is a high symmetry point in reciprocal space and is located at (0,0,0). It may be used as the only sampling point in some DFT calculations.

The Monkhorst-Pack grid is a special set of k-points. They are distributed evenly in reciprocal space and may be aligned such that either one point coincides with Γ or such that 8 points closest points surround Γ at an

Pseudo-potential	Element	a/angs	B_0/GPa
Experimental	Na	4.29[175]	6.3[175]
Na.pz-spn-kjpaw_psl.1.0.0	Na	4.06	8.72
Na.pbe-spn-kjpaw_psl.1.0.0	Na	4.20	7.67
Na.pbessol-spn-kjpaw_psl.1.0.0	Na	4.17	7.50
Experimental	Al	4.05[176]	76[176]
Al.pz-nl-kjpaw_psl.1.0.0	Al	3.98	78.6
Al.pz-nl-kjpaw_psl.1.0.0	Al	3.98	78.6
Al.pbe-nl-kjpaw_psl.1.0.0	Al	4.04	91.3
Al.pbe-nl-kjpaw_psl.1.0.0	Al	4.04	75.0
Al.pbessol-nl-kjpaw_psl.1.0.0	Al	4.01	87.7
Al.pbessol-nl-kjpaw_psl.1.0.0	Al	4.01	79.0

Table 5.7: Experimental vs LSDA vs GGA - the DFT values were computed with a PWscf[177] using a 2x2x2 cell, 7x7x7 kpoints and $\text{ecutwfc}=50$

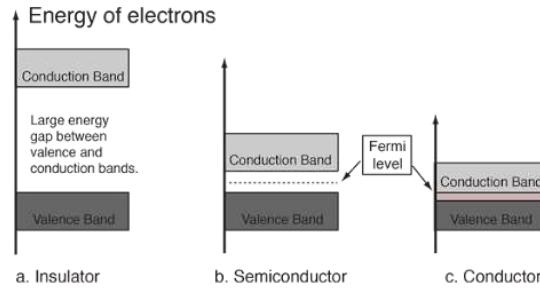


Figure 5.20: Energy gap between valence and conduction bands[181]

equal distance. By offsetting the grid from Γ and the coordinate axis of reciprocal space, the sampling better avoids points of high symmetry.

Where the system is spin degenerate, each state is occupied by 2 electrons for the lower $\frac{N_e}{2}$ states ($f(\vec{k}, i) = 2$), and zero otherwise ($f(\vec{k}, i) = 0$)[180]. This drop, from states occupied by two electrons to empty states occurs at the Fermi surface and it results in a discontinuity in the functions being integrated over in eq. 5.137. This is the case for an electron gas at absolute zero, but if the gas is heated then there is sufficient energy for electrons to occupy states above the Fermi energy. This is a problem that affects metals due to the overlap in conduction and valence bands (fig. 5.20).

Fermi-Dirac statistics determine the probability of a Fermion having an energy E (eq. 5.138). As the temperature increases the probability of finding a fermion above the Fermi energy increases (fig. 5.21). This also smooths the discontinuity at the Fermi surface between the occupied and unoccupied states (at 0K).

$$F(E) = \frac{1}{\exp\left(\frac{E-E_F}{kT}\right) + 1} \quad (5.138)$$

To avoid having to integrate the BZ with a very fine mesh, a smearing function is introduced to remove the discontinuity at the Fermi surface, making the integral over the BZ differentiable at every point. The relationship between the delta and Heaviside step is used to replace the step function when integrated (eq. 5.139).

$$\int_{-\infty}^{\infty} \delta(k) dk = \Theta(x) \quad (5.139)$$

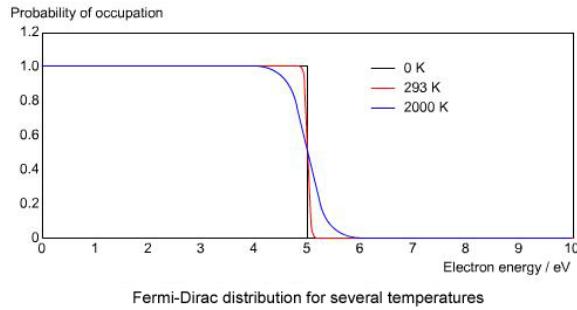


Figure 5.21: Probability $F(E)$ of finding a fermion with energy E at several temperatures[182]

$$I = \int_{-\infty}^{\infty} S(\epsilon - E_F) \int_{BZ} f(\vec{k}) \delta(\epsilon - E(\vec{k})) d\vec{k} d\epsilon \quad (5.140)$$

The step function at the Fermi surface (eq. 5.140), that causes the discontinuity, may be replaced with a smearing function[183]. Four of these functions are available to use in the PWscf DFT code.

The Fermi-Dirac function, as mentioned above, is used to heat the system. It removes the discontinuity but the SCF will converge to the wrong energy (eq. 5.141).

$$S_{FD}(\epsilon) = \frac{1}{\exp\left(\frac{\epsilon - E_F}{kT}\right) + 1} \quad (5.141)$$

A Gaussian smear is also used to replace the step function although, unlike the Fermi-Dirac function, it has no physical meaning (eq. 5.142).

$$S_G(\epsilon) = \frac{1}{2} \left[1 - \operatorname{erf} \left(\frac{\epsilon - \mu}{\sigma} \right) \right] \quad (5.142)$$

The Methfessel-Paxton method attempts to correct the errors introduced by the previous smearing functions where the delta function is replaced with Hermite polynomials (fig. 5.22 and eq. 5.144). A drawback of this method is that it allows for negative occupancy of electron states[180].

$$x = \epsilon - E_F \quad S_{MP,N}(x) = \frac{1}{2} [1 - \operatorname{erf}(x)] + \sum_{n=1}^N A_n H_{2n+1}(x) \exp(-x^2) \quad A_n = \frac{(-1)^n}{n! 4^n \sqrt{\pi}} \quad (5.143)$$

The Mazari-Vanderbilt function was developed to address the short falling of the Methfessel-Paxton function. The occupancy where this function is used is always positive (fig. 5.23).

$$x = \frac{\mu - \epsilon}{\sigma} \delta(x) = \frac{1}{\sqrt{\pi}} \exp\left(-(x - (\frac{1}{\sqrt{2}}))^2\right) (2 - \sqrt{2}x) \quad (5.144)$$

The recommended smearing function for Quantum Espresso is the Mazari-Vanderbilt type. Dr Mazari who co-developed the smearing type also contributed to the development of Quantum Espresso. Dr Vanderbilt has researched condensed-matter physics for 30 years and an entire category of pseudopotentials is named after him.

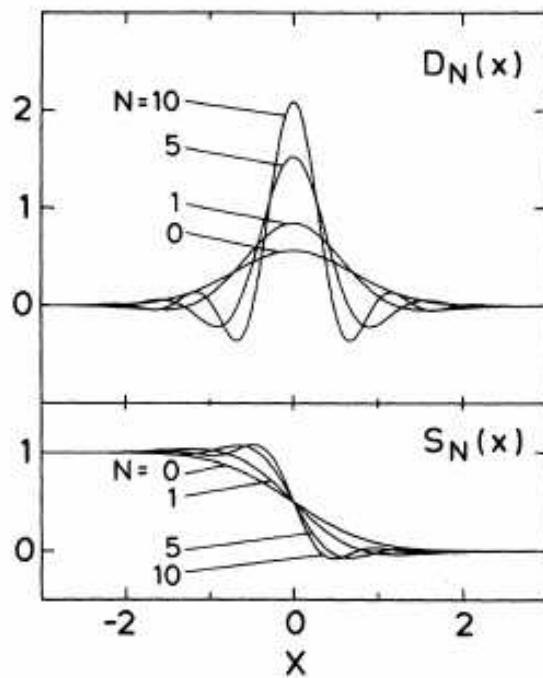


Figure 5.22: Delta function replaced by successive Hermite polynomials[183]

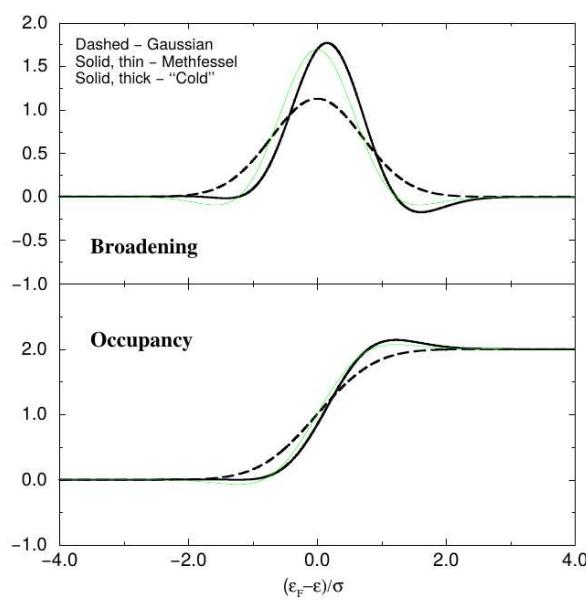


Figure 5.23: The occupancy for the Marzari-Vanderbilt smearing function is always positive, unlike the Methfessel-Paxton function[184]

5.7.13 Ferromagnetism, Antiferromagnetism and Hund's Rule

Electrons are spin-half particles. There isn't a classical analogue to spin, but it may (incorrectly) be imagined as an electron spinning and creating a magnetic field. The electrons may also be (incorrectly) pictured as orbiting the nucleus. This combination of orbital and intrinsic momentum give the total electronic angular momentum $\vec{J} = \vec{L} + \vec{S}$, and the total magnetic dipole moment is proportional to this.

Electrons fill the shells of a ground state atom such that the energy is minimised. Four quantum numbers are used to describe electrons bound to an atom: n , l , m_l and m_s . The Pauli exclusion principle states that the electrons, which are fermions, cannot have the same quantum numbers as another electron bound to that atom.

Electrons may be spin up or spin down thus two electrons can have the same n , l , m_l with two values for m_s available. The energy is minimised by not pairing up and down electrons until all the free slots due to the coulomb interaction between the electrons. Once each slot contains one electron, they begin to pair.

The electronic configuration of Fe highlights this (eq 5.145).

Electronic configuration of Iron

$$\begin{array}{ll}
 1s^2 & \underline{\uparrow\downarrow} \\
 2s^2 & \underline{\uparrow\downarrow} \\
 2p^6 & \underline{\uparrow\downarrow} \quad \underline{\uparrow\downarrow} \quad \underline{\uparrow\downarrow} \\
 3s^2 & \underline{\uparrow\downarrow} \\
 3p^6 & \underline{\uparrow\downarrow} \quad \underline{\uparrow\downarrow} \quad \underline{\uparrow\downarrow} \\
 4s^2 & \underline{\uparrow\downarrow} \\
 3d^6 & \underline{\uparrow\downarrow} \quad \underline{\uparrow} \quad \underline{\uparrow} \quad \underline{\uparrow} \quad \underline{\uparrow}
 \end{array} \tag{5.145}$$

Rather than fill the 3d shell from left to right, the first five slots take electrons with spin in the same direction, and the remaining electron fills the first slot with spin opposite to the first five electrons.

Being fermions the wavefunction for the two electrons must be antisymmetric, with spins opposite to one another. Electrons with the same spin will repel one another and this causes an increase in the screening between the electrons and the nucleus. The screening lowers the attraction between the electrons and the nucleus that then results in the total energy of the atom decreasing[185].

Electronic configuration of Chromium (outer shells)

$$\begin{array}{ll}
 4s^1 & \underline{\uparrow} \\
 3d^5 & \underline{\uparrow} \quad \underline{\uparrow} \quad \underline{\uparrow} \quad \underline{\uparrow} \quad \underline{\uparrow}
 \end{array} \tag{5.146}$$

Electronic configuration of Iron (outer shells)

$$\begin{array}{ll}
 4s^2 & \underline{\uparrow\downarrow} \\
 3d^6 & \underline{\uparrow\downarrow} \quad \underline{\uparrow} \quad \underline{\uparrow} \quad \underline{\uparrow} \quad \underline{\uparrow}
 \end{array} \tag{5.147}$$

Electronic configuration of Cobalt (outer shells)

$$\begin{array}{ll}
 4s^2 & \underline{\uparrow\downarrow} \\
 3d^7 & \underline{\uparrow\downarrow} \quad \underline{\uparrow\downarrow} \quad \underline{\uparrow} \quad \underline{\uparrow} \quad \underline{\uparrow}
 \end{array} \tag{5.148}$$

Electronic configuration of Nickel (outer shells)

$$\begin{array}{c} 4s^2 \quad \underline{\uparrow\downarrow} \\ 3d^8 \quad \underline{\uparrow\downarrow} \quad \underline{\uparrow\downarrow} \quad \underline{\uparrow\downarrow} \quad \uparrow \quad \uparrow \end{array} \quad (5.149)$$

In pure Fe (eq. 5.147), under normal conditions, it is energetically favourable for the atoms to align in the same direction, and as such Fe is ferromagnetic. Co (eq. 5.148) has a face centered tetragonal structure at room temperature. It has a similar electronic structure to iron, but has three unpaired electrons in the 3d shell rather than four. It is a ferromagnetic whilst it remains in its CPH form. Ni (eq. 5.149) is also ferromagnetic, but it is ferromagnetic in its FCC phase.

In its gamma phase, as it is in austenitic stainless steel, Fe atoms align opposite to one another in an antiferromagnetic arrangement. This is also the same for BCC Cr (eq. 5.146) which is antiferromagnetic in its pure form under normal conditions.

5.7.14 Collinear and Non-Collinear DFT Calculations

Although briefly discussed by Kohn and Sham in their 1965 paper[162], DFT does not differentiate between the spins of electrons. However, where the material is in a magnetic field, or where there are unpaired electrons, spin-DFT may be used. This is a good way to model magnetic materials, such as iron, where Hund's rule leads to unpaired electrons as discussed in section 5.7.13.

In collinear calculations the spin of electrons is restricted to two directions, up and down (in the z-axis, in the case of Quantum Espresso), whereas noncollinear calculations give the freedom to orient in any direction.

$$\left(-\frac{1}{2}\nabla^2 + v_{n-e}(\vec{r}_e, \vec{r}_n) + \int d^3\vec{r}' \frac{\rho(\vec{r}'_e)}{|\vec{r}'_e - \vec{r}_e|} + v_{xc}[\rho](\vec{r}_e) \right) \psi_i = E_i \psi_i \quad (5.150)$$

The Kohn-Sham equation is modified to include the magnetic field $B(\vec{r})$ and magnetization density $m(\vec{r})$ [186][187]. Starting with the traditional Kohn-Sham equation, eq. 5.150, the potential and densities are replaced with 2x2 matrices and the equation as a whole is split into spin up and spin down.

$$\left(\left(-\frac{1}{2}\nabla^2 + \int d^3\vec{r}' \frac{\rho(\vec{r}'_e)}{|\vec{r}'_e - \vec{r}_e|} \right) \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} + \vec{v}_{n-e}(\vec{r}_e, \vec{r}_n) + v_{xc}[\vec{\rho}](\vec{r}_e) \right) \begin{bmatrix} \psi_i^\uparrow \\ \psi_i^\downarrow \end{bmatrix} = E_i \begin{bmatrix} \psi_i^\uparrow \\ \psi_i^\downarrow \end{bmatrix} \quad (5.151)$$

The equation is modified as shown in eq. 5.151 and, using the Pauli vector, the potential functions (eq. 5.152 and eq. 5.153)[188] and density functions are rewritten (eq. 5.154)[188].

$$\vec{\rho}(\vec{r}) = \frac{1}{2} \begin{bmatrix} \rho(\vec{r}) + m_z(\vec{r}) & m_x(\vec{r}) - im_y(\vec{r}) \\ m_x(\vec{r}) + im_y(\vec{r}) & \rho(\vec{r}) - m_z(\vec{r}) \end{bmatrix} \quad (5.152)$$

$$\vec{v}(\vec{r}) = \begin{bmatrix} v(\vec{r}) + \mu_B B_z(\vec{r}) & \mu_B(B_x(\vec{r}) - iB_y(\vec{r})) \\ \mu_B(B_x(\vec{r}) + iB_y(\vec{r})) & v(\vec{r}) - \mu_B B_z(\vec{r}) \end{bmatrix} \quad (5.153)$$

$$\vec{v}_{xc}(\vec{r}) = \begin{bmatrix} v_{xc}(\vec{r}) + \mu_B B_{xc,z}(\vec{r}) & \mu_B(B_{xc,x}(\vec{r}) - iB_{xc,y}(\vec{r})) \\ \mu_B(B_{xc,x}(\vec{r}) + iB_{xc,y}(\vec{r})) & v_{xc}(\vec{r}) - \mu_B B_{xc,z}(\vec{r}) \end{bmatrix} \quad (5.154)$$

This work is only concerned with collinear calculations, where the spins are aligned up or down with respect to the z-axis. The equation derived by Von Barth and Hedin may be simplified to give the pair of Kohn-Sham equations in eq. 5.155.

$$\begin{aligned} \left(-\frac{1}{2}\nabla^2 + \int d^3\vec{r}' \frac{\rho(\vec{r}'_e)}{|\vec{r}_e - \vec{r}'_e|} + v_{xc}^\uparrow[\vec{\rho}](\vec{r}_e) + B_z(\vec{r}) \right) \psi_i^\uparrow(\vec{r}) &= E_i^\uparrow \psi_i^\uparrow(\vec{r}) \\ \left(-\frac{1}{2}\nabla^2 + \int d^3\vec{r}' \frac{\rho(\vec{r}'_e)}{|\vec{r}_e - \vec{r}'_e|} + v_{xc}^\downarrow[\vec{\rho}](\vec{r}_e) - B_z(\vec{r}) \right) \psi_i^\downarrow(\vec{r}) &= E_i^\downarrow \psi_i^\downarrow(\vec{r}) \end{aligned} \quad (5.155)$$

This model requires self consistently solving for both the charge density $\rho(\vec{r})$ and magnetisation density $m_z(\vec{r})$. It will be adequate for modelling the ferromagnetism of pure iron and antiferromagnetic structure of iron as it exists in austenitic stainless steel. A full non-collinear treatment would no doubt be better, when considering surfaces and the presence of doped PGMs atoms in the structure, but would require more resources plus reliable and tested pseudopotentials suitable for these calculations.

Chapter 6

Predicting Activation in Proton Irradiated Samples

A computer program, Activity, was developed to calculate the radioactivity of an ion irradiated target. The code has been released in two versions, with the first being written in Fortran and the second in Python. SRIM, an ion transport code, was used to calculate proton trajectories used by the Activity codes.

Decay equations were derived to calculate the activity of isotopes within the target at time t during and after irradiation. These were created in a similar manner to the Bateman equations for radioactive decay, but include source terms and branching factors for radioactive isotopes in a decay chain. These were implemented in a module and class for each version of the code.

The University of Birmingham cyclotron was used to activate Iron samples (this work) and Molybdenum (a colleague's work). The radioactivity of these samples were compared to the predicted activity and gamma lines of the Activity code.

Finally, the latest version of the Activity code was used to predict the activity of a Steel specified by another colleague and a set of Iron samples irradiated to 100DPA at varying proton energies. The results of these simulations are presented in this chapter.

6.1 Introduction

6.1.1 Motivation for an Ion Induced Activity Code

High flux neutron reactors are expensive to use, whereas proton accelerators capable of producing beams of adequate fluence and energy are more readily available, cheaper to buy and cheaper to run. Ion beams, due to the Coulomb interaction, are more controllable in terms of energy, direction and fluence. They may be concentrated on a desired target at a fixed flux and energy for a set amount of time.

Depending on the energy of the ion beam, the target material will become radioactive. The stable nuclei are transmuted and when the resulting isotope is unstable it will decay. An equation was derived to predict the activity of isotopes for any decay chain and this includes source terms and branching factors. Two versions of a computer code were developed to compute the reaction rates and activity of the irradiated targets[103].

At the start of this work, the standard inventory code for computing activity was FISPACT-2007[189]. This takes a spectra and collapses it with the cross section, assuming the irradiation of an infinite and homogenous material. There is no notion of transport in this code. An alumni of the University of Birmingham developed an interface between Monte Carlo N-Particle and FISPACT, known as Fusion Activation Transport Interface (FATI)[190], in order to bring the spatial aspect of MCNP and temporal aspect of FISPACT together for neutron radiation. The latest version of FISPACT (FISPACT-II[191]) was released in 2017, and it uses a sparse matrix to calculate the activity.

Other solvers for Bateman equations exist, but it would be difficult to compare the methods of proprietary code without access to the source code. Branching is a feature added to many methods of solving, but the source term is neglected in discussions[192]. One reason for this may be the isotopes of interest reach saturation when irradiated for long periods in high flux reactors. Ion irradiation has the potential to cause more damage and reduce the irradiation time, not allowing isotopes to reach saturation. In this situation the source and decay rates are important to model throughout the irradiation process.

For codes such as FISPACT there is more emphasis on neutron activation through the development of FATI. The work presented here is interested in radioactive isotopes generated as a result of ion irradiation.

Given the above, the equations and code developed here fill a gap in several ways. Rather than use a sparse matrix based Bateman solver, an equation was derived for both stable and unstable isotopes to compute the amount of an isotope individually with a single function call. Each function call is only dependent upon the input parameters: isotope half lives, production rates, branching factors and starting amounts.

A high resolution database of proton reaction cross sections was created to cover the entire range of energies available on the University of Birmingham Cyclotron, extending up to 50MeV in increments of 10keV. This database is coupled to proton trajectory data from SRIM and the single function call decay equations.

Combined, this allows the user to predict the radioactivity of an ion irradiated target where the decay equations allow isotopes to decay away during and after irradiation. It makes no difference to the code whether or not the isotopes created reach saturation.

The code allows the user to visualise the build up and decay of isotopes over time during and after irradiation. This is in the form of plots and several videos output by the code. It also allows the user to identify isotopes of particular interest where a complicated material, containing many isotopes, is due to be irradiated.

6.1.2 Summary of Chapter

In this chapter the development of the modified Bateman equations will be discussed (6.2). The Activity code and cross section database are introduced (6.3) and this is followed by a short discussion of the SRIM code used to generate trajectory data (6.4). The measured gamma activity following the irradiation of Fe with the University of Birmingham cyclotron (6.5) is compared with predictions by the activity code; the predictions are extended to 100DPA irradiated iron samples for a range of proton energies (6.6). Finally the measured results following the irradiation of Mo are compared to those computed using the code (6.7 and a complex Mo doped steel is analysed (6.8) to investigate expected radiation levels when irradiated with 5MeV protons.

6.2 Activation by Ion Irradiation

The Bateman equation was derived using Laplace transforms, and this same method has been used to develop a modified equation that incorporates branching factors and production rates for each isotope in the decay chain, as illustrated by fig. 6.1[103].

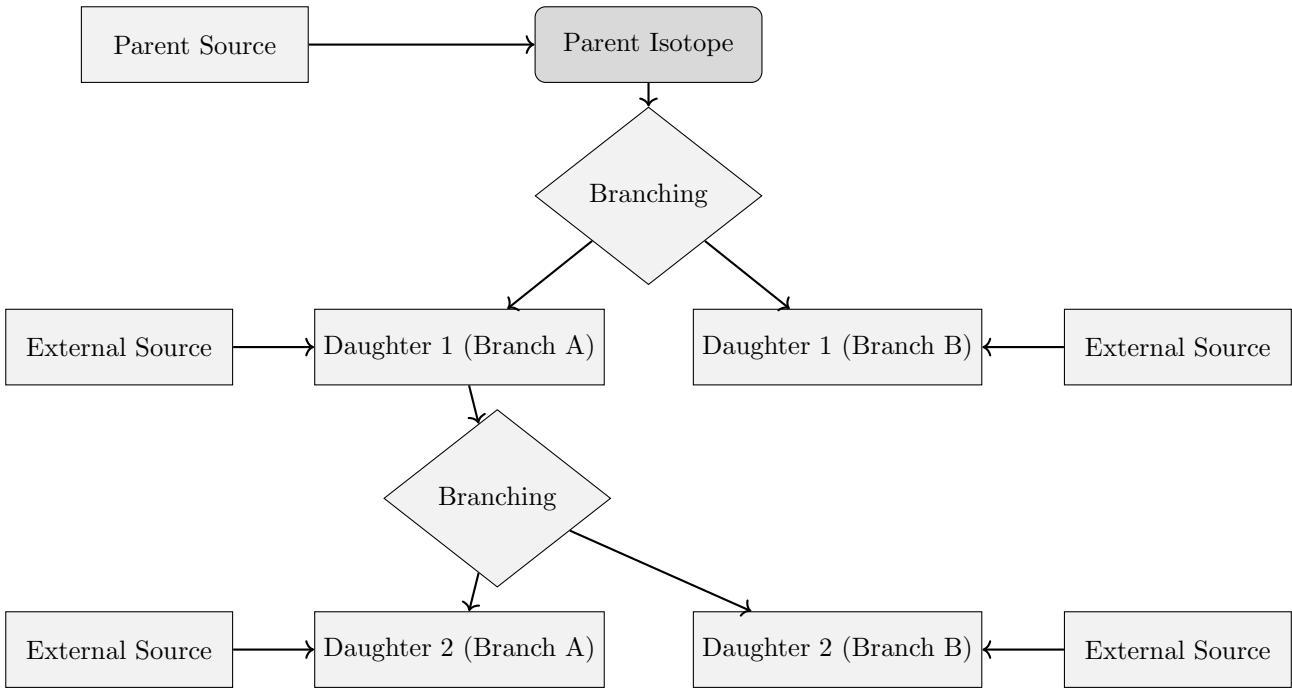


Figure 6.1: An example of several decay chains including branching factors and possible external source terms for each isotope on each chain.

6.2.1 Laplace Transform

Laplace Transforms (eq. 6.1) are a useful mathematical tool, and allow ordinary differential equations to be solved by simple algebraic manipulation in the s domain (appendix A). Bateman took advantage of Laplace Transforms in deriving his equation, and this is the method that has been taken here as well.

$$F(s) = \int_0^{\infty} f(t) \exp(-st) dt \quad (6.1)$$

$$f(t) = \frac{1}{2\pi i} \int_{\gamma-i\infty}^{\gamma+i\infty} \exp(st) F(s) ds \quad (6.2)$$

The inverse of the Laplace transform is the Bromwich integral (eq. 6.2), where γ is a vertical contour in the complex plane[193]. The inverse may be found using a table of standard inverses (appendix A) or by numerical methods (6.2.3).

6.2.2 Constructing the Differential Equations

The first step is to set up differential equations for the parent isotope, unstable daughter isotopes and stable daughter isotope. The parent isotope has a source term, due to production, and a loss term, due to decay. The unstable daughter isotopes have two source terms, from the production by irradiation induced transmutation and the decay of preceding isotopes in the decay chain, and a loss term, due to decay. Finally, the stable daughter that finalizes the decay chain has two source terms (the same as the unstable daughters) but no loss term[103].

The variables (and vectors) used in these equations are defined as follows:

- $\vec{\lambda}$ vector containing isotope decay constants λ_i

- \vec{b} vector containing isotope to isotope branching factors b_i
- \vec{w} vector containing isotope production rates w_i
- t time at which activity/amount of isotope is measured
- $N_i(0)$ starting amount of the i^{th} isotope
- $N_i(t)$ amount of the i^{th} isotope at time t
- $N'_i(t)$ change in amount of the i^{th} isotope, with respect to time, at time t

The differential equations for the parent isotope (first isotope), unstable daughter isotopes (i^{th} isotopes) and stable, final, daughter isotope (z^{th} isotope) in the time domain are as follows:

$$N'_1(t) = \omega_1 - \lambda_1 N_1(t) \quad (6.3)$$

$$N'_i(t) = \omega_i + b_{i-1} \lambda_{i-1} N_{i-1}(t) - \lambda_i N_i(t) \quad (6.4)$$

$$N'_z(t) = \omega_z + b_{z-1} \lambda_{z-1} N_{z-1}(t) \quad (6.5)$$

Applying the Laplace Transform to these three differential equations allows them to be manipulated and solved algebraically in the s-domain.

$$N_1(s) = \frac{1}{s + \lambda_1} N_1(0) + \frac{1}{s(s + \lambda_1)} \omega_1 \quad (6.6)$$

$$N_i(s) = \frac{1}{s(s + \lambda_i)} (\omega_i) + \frac{1}{s + \lambda_i} (b_{i-1} \lambda_{i-1} N_{i-1}(s)) + \frac{1}{s + \lambda_i} N_i(0) \quad (6.7)$$

$$N_z(s) = \frac{1}{s^2} \omega_z + \frac{1}{s} b_{z-1} \lambda_{z-1} N_{z-1}(s) + \frac{1}{s} N_z(0) \quad (6.8)$$

6.2.3 Numerical Inversion of the Laplace Transform

The Gaver-Stehfest[194] algorithm was developed in the 1960s and 1970s and is a method of calculating the inverse of a Laplace Transform in the real number domain. It is an easy to implement and reasonably accurate method, although it is an approximation to the real value. A comparison between an analytic and numeric inversion for the unstable isotope Po-218 is discussed at the end of this section (fig. 6.2).

$$f(t) \approx f_n(t) = \frac{\ln(2)}{t} \sum_{k=1}^{2n} a_k(n) F(s) \text{ where } n \geq 1, t > 0 \quad (6.9)$$

$$s = \frac{k \ln(2)}{t} \quad (6.10)$$

$$a_k(n) = \frac{(-1)^{(n+k)}}{n!} \sum_{j=\text{Floor}(\frac{k+1}{2})} j^{n+1} \binom{n}{j} \binom{2j}{j} \binom{j}{k-j} \quad (6.11)$$

The equation for the i^{th} isotope may be calculated by recursively calculating the equations by numeric inversion, starting from the first (parent isotope) and inserting the result into each subsequent recursion until the i^{th} isotope is reached (changing the equations appropriately for the parent, unstable daughter and stable daughter isotopes).

6.2.4 Analytic Solution by Partial Fraction Expansion

The equation for the i^{th} isotope in the s domain can be written in full by substituting the preceding equation until the parent isotope is reached, and this full equation may be rearranged with the production amount of each isotope and starting amount of each isotope in individual terms. Each of these terms is multiplied by a fraction that can be expanded, using partial fractions, and inverted analytically.

This is illustrated with an example unstable isotope, fourth in the decay chain (including the parent isotope) (eq. 6.12).

$$\begin{aligned}
 N_4(s) = & \frac{1}{(s + \lambda_1)(s + \lambda_2)(s + \lambda_3)(s + \lambda_4)} b_2 b_3 b_4 \lambda_1 \lambda_2 \lambda_3 N_1(0) \\
 & + \frac{1}{(s + \lambda_2)(s + \lambda_3)(s + \lambda_4)} b_3 b_4 \lambda_2 \lambda_3 N_2(0) \\
 & + \frac{1}{(s + \lambda_3)(s + \lambda_4)} b_4 \lambda_3 N_3(0) \\
 & + \frac{1}{(s + \lambda_4)} N_4(0) \\
 & + \frac{1}{s(s + \lambda_1)(s + \lambda_2)(s + \lambda_3)(s + \lambda_4)} b_2 b_3 b_4 \lambda_1 \lambda_2 \lambda_3 \omega_1 \\
 & + \frac{1}{s(s + \lambda_2)(s + \lambda_3)(s + \lambda_4)} b_3 b_4 \lambda_2 \lambda_3 \omega_2 \\
 & + \frac{1}{s(s + \lambda_3)(s + \lambda_4)} b_4 \lambda_3 \omega_3 \\
 & + \frac{1}{s(s + \lambda_4)} \omega_4
 \end{aligned} \tag{6.12}$$

An example stable isotope, fourth (last) in the decay chain (including the parent isotope) (eq. 6.13).

$$\begin{aligned}
 N_4(s) = & \frac{1}{s(s + \lambda_1)(s + \lambda_2)(s + \lambda_3)} b_2 b_3 b_4 \lambda_1 \lambda_2 \lambda_3 N_1(0) \\
 & + \frac{1}{s(s + \lambda_2)(s + \lambda_3)} b_3 b_4 \lambda_2 \lambda_3 N_2(0) \\
 & + \frac{1}{s(s + \lambda_3)} b_4 \lambda_3 N_3(0) \\
 & + N_4(0) \\
 & + \frac{1}{s^2(s + \lambda_1)(s + \lambda_2)(s + \lambda_3)} b_2 b_3 b_4 \lambda_1 \lambda_2 \lambda_3 \omega_1 \\
 & + \frac{1}{s^2(s + \lambda_2)(s + \lambda_3)} b_3 b_4 \lambda_2 \lambda_3 \omega_2 \\
 & + \frac{1}{s^2(s + \lambda_3)} b_4 \lambda_3 \omega_3 \\
 & + \frac{1}{s^2} \omega_4
 \end{aligned} \tag{6.13}$$

By using partial fraction expansion and standard Laplace Transforms, the set of equations (eqs. 6.14, 6.15,

6.16, 6.17) is used to calculate the amount of the m^{th} isotope in the decay chain, providing the m^{th} isotope is unstable[103].

$$N_m(t; \vec{\lambda}, \vec{b}, \vec{w}, \vec{N}_0) = \sum_{k=1,m} r(k, m, \vec{\lambda}, \vec{b}) [f(t; k, m, \vec{\lambda}) N_{0,k} + g(t; k, m, \vec{\lambda}) w_k] \quad (6.14)$$

$$r(k, m, \vec{\lambda}, \vec{b}) = \begin{cases} \prod_{i=k, m-1} (b_{i+1} \lambda_i), & \text{if } k < m \\ 1, & \text{if } k = m \end{cases} \quad (6.15)$$

$$f(t; k, m, \vec{\lambda}) = (-1)^{m-k} \sum_{i=k, m} \left[\frac{\exp(-\lambda_i t)}{\prod_{j=k, m; j \neq i} (\lambda_i - \lambda_j)} \right] \quad (6.16)$$

$$g(t; k, m, \vec{\lambda}) = \frac{1}{\prod_{i=k, m} \lambda_i} + (-1)^{m-k+1} \sum_{i=k, m} \left[\frac{\exp(-\lambda_i t)}{\lambda_i \prod_{j=k, m; j \neq i} (\lambda_i - \lambda_j)} \right] \quad (6.17)$$

The set of equations (eqs. 6.18, 6.19, 6.20, 6.21) is used to calculate the amount of the m^{th} isotope in the decay chain, where the m^{th} isotope is stable[103].

$$N_m(t; \vec{\lambda}, \vec{b}, \vec{w}, \vec{N}_0) = N_m + w_m t + \sum_{k=1, m-1} r(k, m, \vec{\lambda}, \vec{b}) [f(t; k, m-1, \vec{\lambda}) N_{0,k} + g(t; k, m, \vec{\lambda}) w_k] \quad (6.18)$$

$$r(k, m, \vec{\lambda}, \vec{b}) = \begin{cases} \prod_{i=k, m-1} (b_{i+1} \lambda_i), & \text{if } k < m \\ 1, & \text{if } k = m \end{cases} \quad (6.19)$$

$$f(t; k, m, \vec{\lambda}) = \frac{1}{\prod_{i=k, m-1} \lambda_i} + (-1)^{m-k+1} \sum_{i=k, m-1} \left[\frac{\exp(-\lambda_i t)}{\lambda_i \prod_{j=k, m-1; j \neq i} (\lambda_i - \lambda_j)} \right] \quad (6.20)$$

$$g(t; k, m, \vec{\lambda}) = \frac{1}{\prod_{i=k, m-1} \lambda_i} t - \frac{\sum_{i=k, m-1} [\prod_{j=k, m-1; j \neq i} \lambda_j]}{\prod_{i=k, m-1} \lambda_i^2} + (-1)^{m-k+1} \sum_{i=k, m-1} \left[\frac{\exp(-\lambda_i t)}{\lambda_i^2 \prod_{j=k, m-1; j \neq i} (\lambda_i - \lambda_j)} \right] \quad (6.21)$$

6.2.5 Preference: Analytic over Numeric

The numeric solution only requires the equation to be solved in the s-domain; the Gaver-Stehfest algorithm performs the inversion. It is worth the extra effort to derive and implement an analytic solution, as the numeric is only an approximation. Examples of the pitfalls of the numeric solution are that it can give negative amounts of an isotope and the difference between the numeric and analytic calculated amounts can become quite large

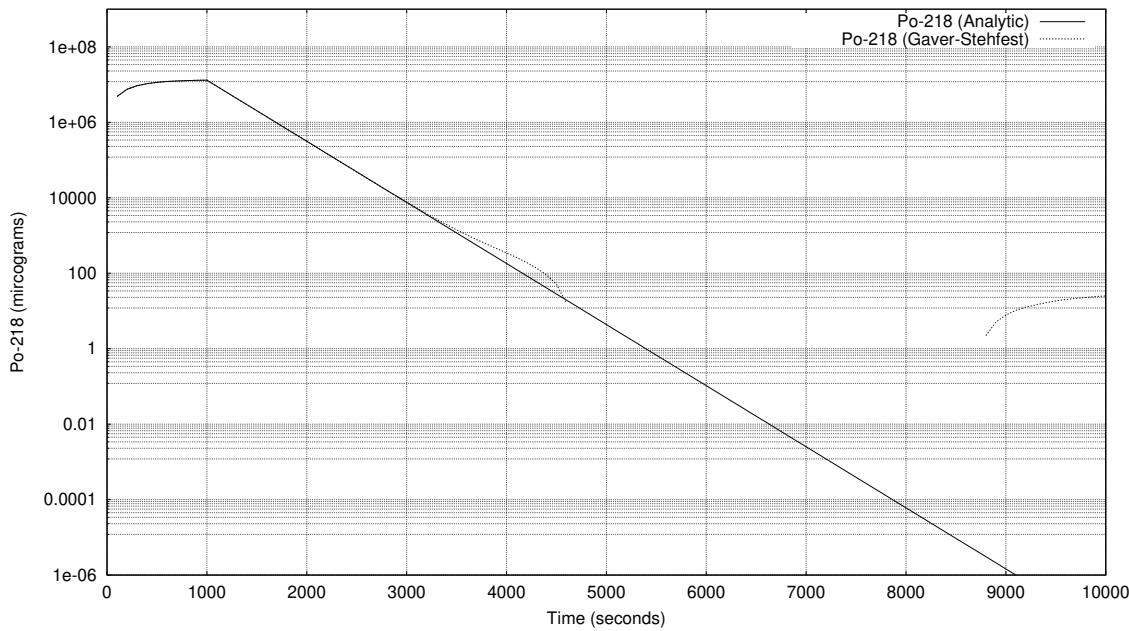


Figure 6.2: Decay of Po-218: Analytic and Gaver-Stehfest Calculations [26]

when the isotope decays away to a very small value. Fig. 6.2[103] shows the predicted decay of a sample of Po-218 irradiated for 1,000s, and sampled until 10,000s. In the region between 4,000s and 9,000s the amount from the numeric calculation drops below zero, whereas the analytic calculation remains above zero, as would be expected.

6.3 Activity Code

6.3.1 Computer Code Development

The first version of the Activity code was written in Fortran. A paper that describes this code is contained in section C.1.1 of the appendix along with the manual describing how to use it (section C.1.1). This version used the TENDL-2013 cross section database and was replaced by a version of the code written in Python.

Activity V2 uses a custom cross section database generated using the TALYS nuclear reaction code along with the JEFF3.1.1 datafile for decay data. All the data has been processed and is packaged with the source code. The source code and instructions on how to use the program are available to download from GitHub:

https://github.com/BenPalmer1983/activity_v2

The user provides an ion trajectory file and an input file that specifies beam parameters and target composition. The code may be used for both protons and deuterons, providing the correct trajectory file has been generated. It does also contain neutron reaction cross sections and may be used for very thin foils with a flat distribution of neutron energies.

6.3.2 Extended Bateman Equations

The Bateman equations were derived from scratch to include a source term for each isotope and branching factors from parent isotope to daughter isotope(s). This process along with the numerical and analytic solutions are outlined in section 6.2.4.

The decay equations were programmed as a class in python (appendix B) that used decay data accessible through a second class, isotopes, populated with data from the JEFF 3.1.1 data file. The static function for computing activity in this class was tested against numerically computed values.

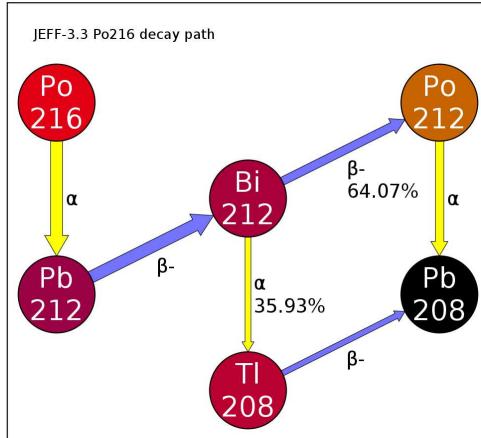


Figure 6.3: Decay path of Polonium-216

Due to the large range in magnitude of the half life for various isotopes, a decay chain was selected where the time step for the numeric calculation would be applicable to the half life of the isotopes in the decay chain, whilst keeping the number of steps low enough to compute the activities in a reasonable amount of time with a simple algorithm. The decay path of $^{216}_{84}Po$ (fig. 6.3) contains 6 isotopes and includes one branch at $^{212}_{83}Bi$.

Isotope	Half Life (s)
Po-216	1.45×10^{-1}
Pb-212	3.83×10^4
Bi-212	3.63×10^3
Po-212	2.99×10^{-7}
Tl-212	3.1×10^1
Pb-208	Stable

Table 6.1: Half life of isotopes in the Po-216 decay chain

The range of half life times (table 6.1) in the Polonium decay chain was from 3.83×10^4 s for $^{212}_{82}Pb$ down to 2.99×10^{-7} s for $^{212}_{83}Po$ and the maximum number of steps used in the numeric calculation is 10^{10} . This number of steps was near the limit at which increasing by another order of magnitude would take a significant amount of time to run the calculation.

Isotope	Source Rate	Starting amount N_0
Po-216	2.0×10^{-1}	1.0×10^2
Pb-212	0.0×10^0	5.0×10^0
Bi-212	7.0×10^{-2}	1.5×10^1
Po-212	5.0×10^{-3}	0.0×10^0
Tl-212	0.0×10^0	0.0×10^0
Pb-208	1.0×10^{-2}	3.0×10^2

Table 6.2: Parameters used for the decay equation vs numeric calculation comparison

The numeric code used to perform these calculations is detailed in the appendix (section B.3.1). The calculation was set up with a range of source rates and isotope starting amounts including no source for $^{212}_{81}Tl$ and a zero starting amount for both $^{212}_{81}Tl$ and $^{212}_{83}Po$. The full details are given in table 6.2 and these were used in both the numeric and analytic codes.

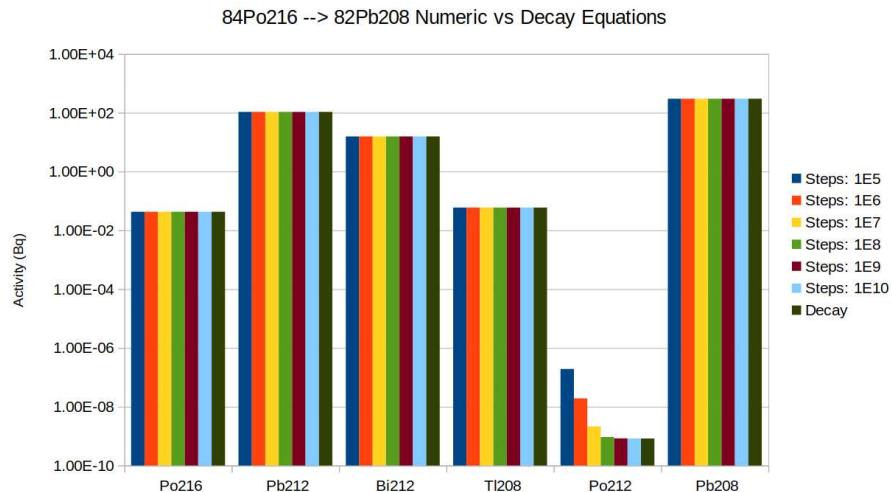


Figure 6.4: Decay path of Polonium-216

As seen in fig. 6.4, for all but the shortest half life isotope Po-212, the numeric calculations are in good agreement with the decay equations. As the number of steps increases, and hence the time step decreases, the numerically computed activity for Po-212 also converges to that calculated with the decay equations.

The modified Bateman equation was also tested with several other decay chains against a numeric solution for each chain. These were the decay of $^{49}_{24}Cr$, $^{66}_{28}Ni$, $^{125}_{55}Cs$, $^{213}_{83}Bi$, $^{216}_{84}Po$ and $^{218}_{86}Rn$. Apart from a break down in the numeric code, where the half life was comparable to or smaller than the time step, the results from both the equations and the numeric solver were in good agreement. The full results to this testing are included in appendix B.3.1.

6.3.3 Talys Generated Proton Cross Section Database

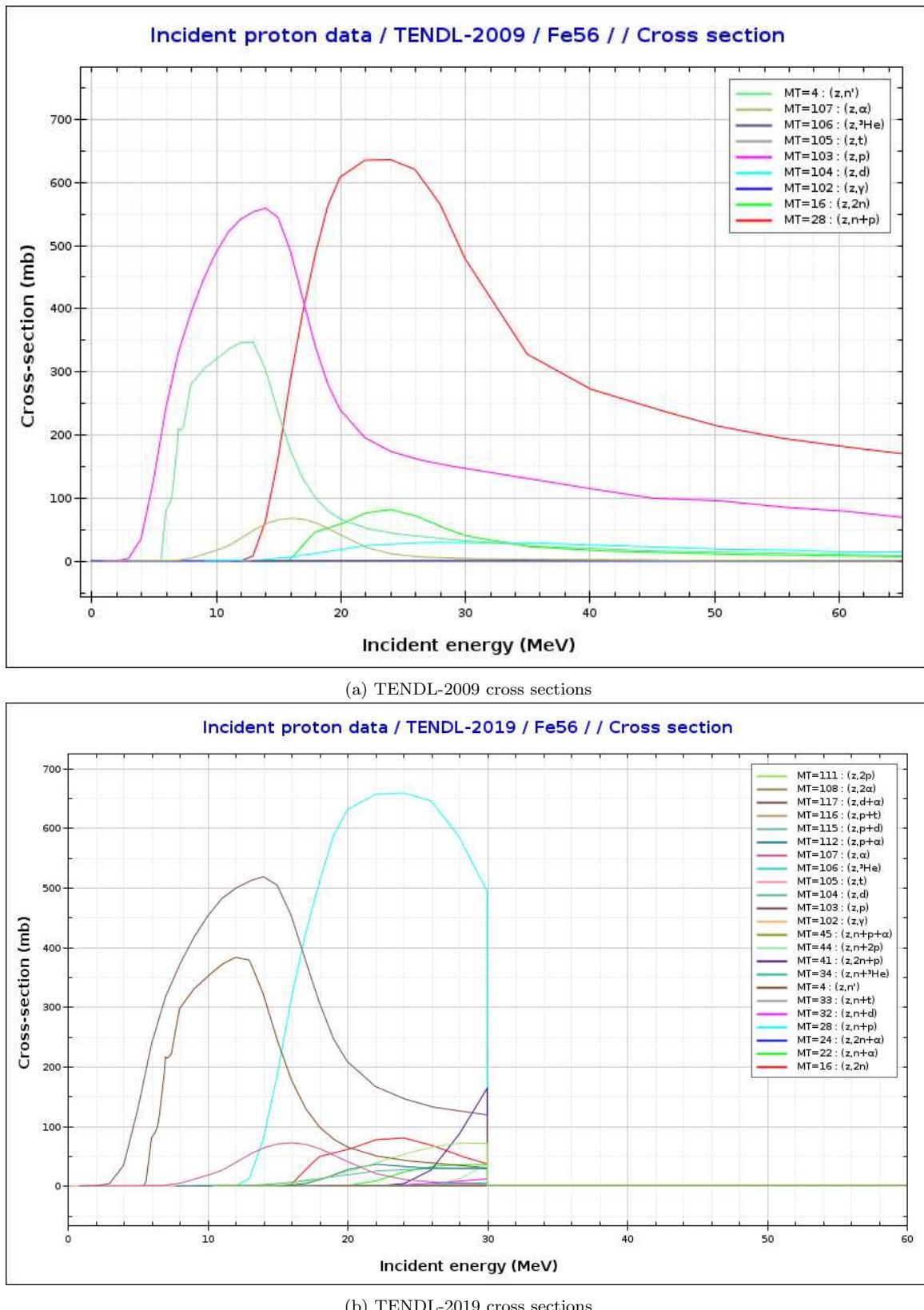
Due to a variation in range of energies in the existing cross section data files, the cross section data was computed from scratch using the TALYS code and the BlueBEAR HPC.

A set of proton reaction cross section data files already exist and are updated every few years. The type and range of data changes between versions of the file. For example, the TENDL-2014 version contains only elastic and (z,anything) files for (p,Fe56). As this reaction (z,anything) is the sum of all the reactions with individual isotopes, the user is unable to determine the reaction rates for those individual isotopes. The TENDL-2009 file contains a wider ranger of reactions and under 50 data points span the energy range from 0MeV to 200MeV. The 2019 version contains even more reactions but spans a much shorter energy range, 0MeV to 28MeV, with cross sections given at just under 60 energy points(fig. 6.5).

The cyclotron used in the experimental work has an energy range up to 40MeV for protons and 53MeV for $^4He^{2+}$ ions. To cover this entire range, the TALYS code was used to compute cross section reaction data from 0.01MeV to 62.0MeV in increments of 10keV, breaking the energy range up into 6,200 data points for each reaction.

Limitations of the TALYS code caused the calculations to be processed in batches of either 400 or 600 points. The data was collated and processed resulting in a datafile of particle production reactions and residual production reactions. This is accessible through a stand-alone python class that is also a part of the Activity V2 code.

The residual reactions are grouped by target isotope and then by residual isotope, whereas the particle cross sections are grouped by target isotope then particle type. The computed values are plotted for all available reactions and all residuals for $^{56}_{26}Fe$ reactions with protons (figs. 6.6) and these show a good agreement, although there is a more detailed set of residual isotopes for the generated cross sections.

Figure 6.5: Data from pre-existing TENDL files for $(^{56}\text{Fe}, p)$ reactions

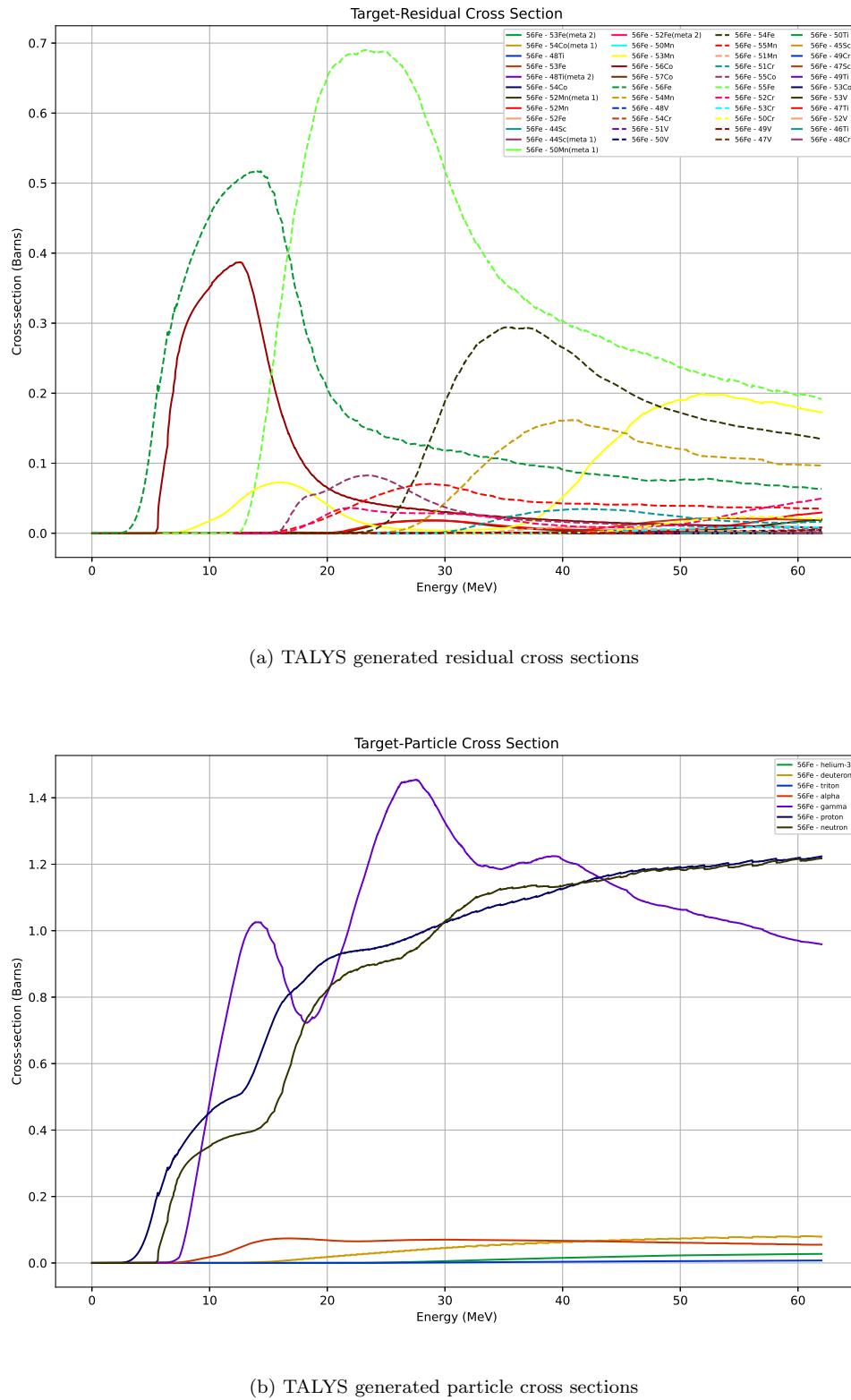


Figure 6.6: Data from TALYS generated files for $(^{56}\text{Fe}, p)$ reactions

Listing 6.1: Status report from TALYS database code

```

1 ######
2      TALYS Cross Section Python Dictionary
3 #####
4
5      DB STATUS
6
7 Talys dir:      ../../data/talys
8 Isotopes file:  ../../data/isotopes.pz
9 Database load time: 24.489s
10
11 Projectiles: n d p
12
13 Total data points: 18777965
14
15 End time: 24.674s

```

The complete set of data files, and Python class used to read them, are available to download at:

https://github.com/BenPalmer1983/talys_xs_db

The cross section data output by TALYS was grouped into four main files: elastic_xs.pz, nonelastic_xs.pz, particle_xs.pz and residual_xs.pz. All data files that make up this database currently total over 400MB and contain over 18 million data points (listing 6.1), although this could increase if more projectiles and a wide range of energies are added.

6.3.4 Ion Trajectory Data

The SRIM code is used to generate ion trajectory data. This uses a quantum mechanical treatment of ion-atom collisions and statistical algorithms to speed up the process[195]. The trajectory data files used by both versions of the Activity code will be created using SRIM for the required beam energy and target combinations.

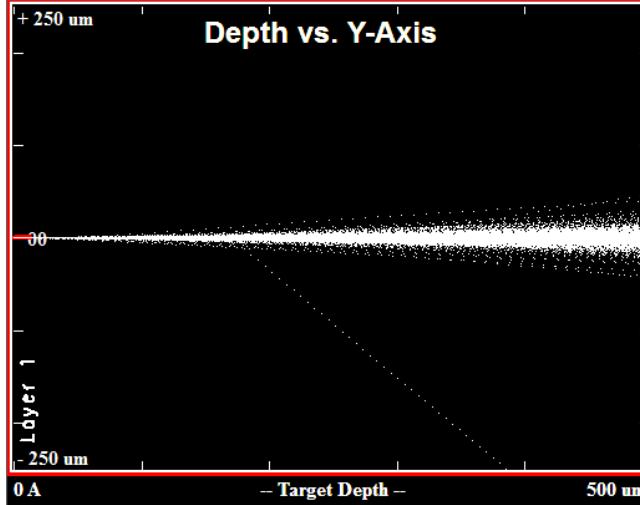


Figure 6.7: 36MeV proton tracks into a 0.5mm thick steel generated with the SRIM code where protons enter the material from the left of the figure and exit on the right.

In the example where the target is set as pure Iron, due to the way SRIM functions, the structure type is inconsequential. At just 0.5mm in thickness, the 36MeV ions pass through the targets and exit with an energy above 30MeV (figure 6.7). The code creates a data file, exyz.txt, and this covers the ion trajectory over this range of ion energies.

6.3.5 Activity Code Assumptions

A number of assumptions have been made to simplify the model. These may be addressed in future versions if development of the code continues. The flux of the beam is assumed to be constant at any depth into the material. Charged particle irradiation is ignored completely from the decay of radioactive isotopes in the target, and the amount of beta radiation may determine how the target is safely handled.

Gammas are assumed to be emitted isotropically and degrade only due to the increase in surface area as the distance from the source increases. No consideration is given to the dispersion of radioactive isotopes through the target material, how much material the gammas must pass through or how much air they must go through in order to reach the person (or measuring device). These assumptions and possible improvements are considered in chapter 9.

6.3.6 Foil Activation by Neutron Irradiation

An additional feature was added to the Activity V2 code to estimate neutron activity based on the existing code developed to calculate ion activation. It is a simple non-transport neutron activation code, used to estimate the activity and subsequent cooling of materials irradiated by neutrons. A single energy neutron flux or Maxwell-Boltzmann distribution may be selected by the user and the TALYS generated database is used to provide cross sections for calculating reaction rates. Finally, the previously derived extended Bateman equations are used to calculate isotopes at time t after irradiation begins.

6.4 Simulating Proton Irradiated Iron with SRIM

The trajectory data used by the activity code, and also to estimate the $^{55}_{27}Co$ activity, was provided by acrshort-srim. An input configuration was set with Iron as the target material and protons as the projectile ion. The sample depth was set to at least the target thickness, 0.5mm. The activity code truncates the data to fit the target thickness, so a much larger depth, say 1m, could be set in SRIM to ensure all the ions stop within the simulated target, and this data file may be reused for targets of varying thicknesses.

The ions travelling through the thin 0.5mm target lose between 4MeV and 5MeV as they pass, and they do so quite smoothly (fig. 6.8). The majority of the energy is lost smoothly through electronic stopping.

The ion trajectory plot shows the deflection of ions in both the y and z axis. A large percentage of the ions here deviate very little from a straight pass through, but several are deflected by quite a large angle. Within the activity computer codes, these ions deflected ions will travel a greater distance through the target. The only cutoffs applied to the ion trajectory data are either the ion reaching 0MeV or travelling more than the depth in the x-axis. Any ions travelling a large distance in the y or z axis are still determined to be within the target.

As the thickness of the iron is increased, all the energy is lost to the iron target, and this loss of energy occurs within a shorter range in the remaining quarter of a millimetre depth of the iron target (fig. 6.9).

6.5 Cyclotron Irradiated Iron

6.5.1 Cyclotron Beam Line

The Scanditronix MC40 cyclotron at the University of Birmingham has several beamlines and is capable of accelerating protons, deuterons, Helium 3 and Helium 4 with fluxes and energy ranges detailed in table 6.3.

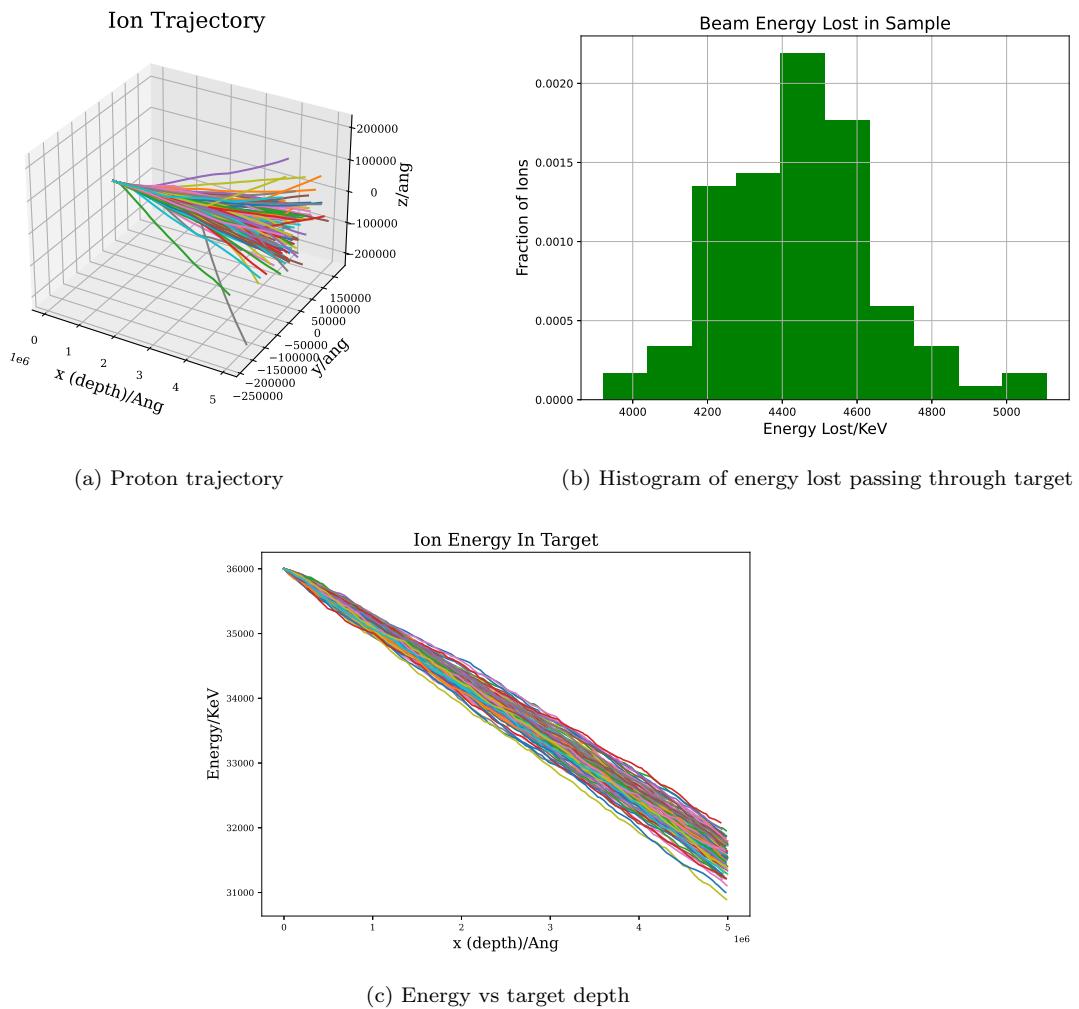


Figure 6.8: Protons through a 0.5mm thick iron target - 100 proton histories

After running a simulation with SRIM, the 36MeV protons were expected to create an average of 15.8 vacancies each.

The cyclotron is located in one room, and several beam lines run from the cyclotron to other rooms in the cyclotron building. The operation room is separate from the cyclotron and beam line rooms (fig. 6.10).

The iron sample was held perpendicular to the beam line, with the ions passing through the 0.5mm thickness of the sheet. The beam area was approximately $6.4 \times 10^{-5} m^2$, irradiating a volume of approximately $3.2 \times 10^{-8} m^3$. The target was irradiated for 300 seconds at 0.5 micro amps and it was expected to cause over 1.4×10^{16} displacements within the volume of iron targetted by the beam. With a number density of approximately 8×10^{28} atoms per cubic meter, giving a relatively low damage dose, when compared to that expected over the lifetime of a component within the reactor, of 5.0×10^{-6} DPA.

During irradiation, the amount of gamma radiation released was enough to trip the alarm for the room (fig. 6.11). The proton fluence was less than 1% of the maximum fluence capable of being produced by the cyclotron, so this was definitely a concern. To increase the damage dose, and keep to a shorter period of time, the current would most probably be increased to a much higher percentage, also increasing the rate of Gammas produced during irradiation.

Preliminary investigations with SRIM into the stopping distance of protons in a range of materials shows 36MeV ions stop within 3mm in Iron and Steel (fig. 6.12).

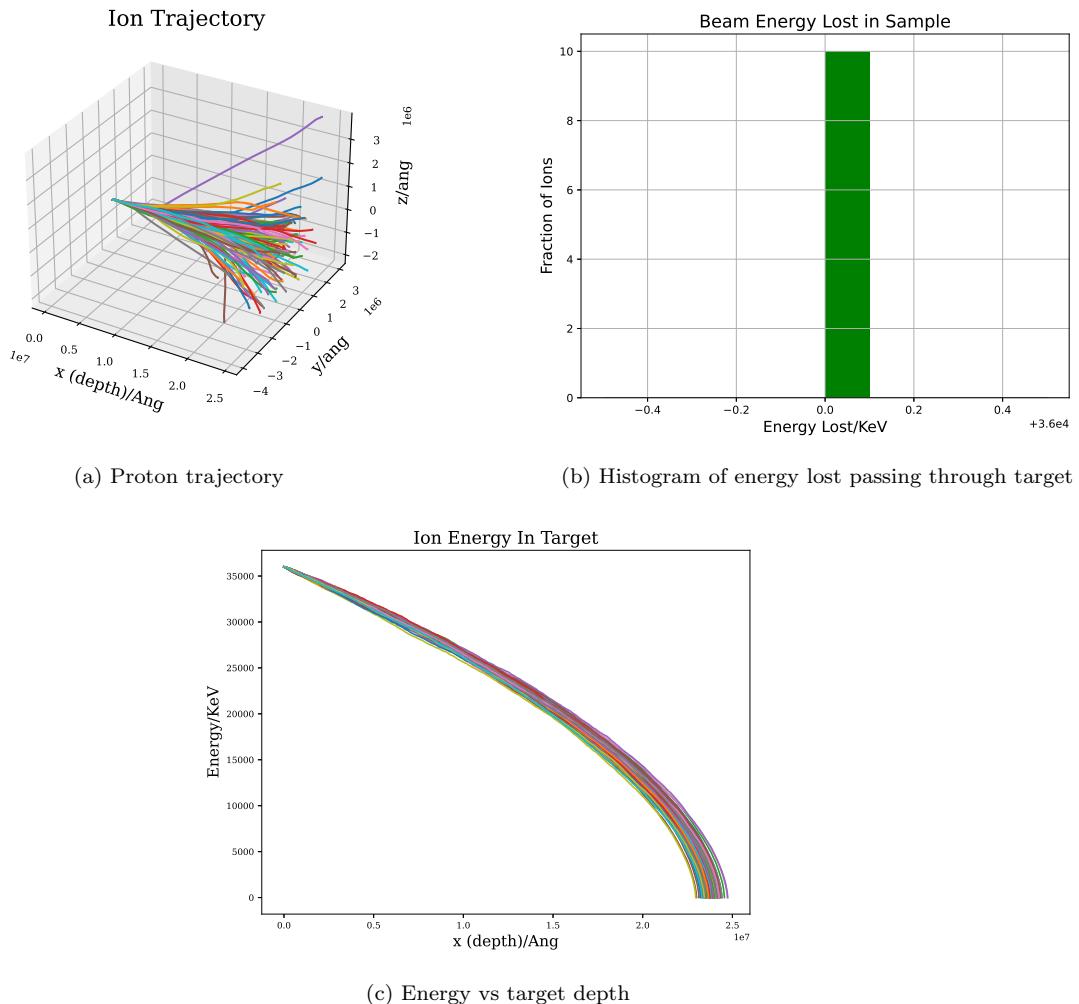


Figure 6.9: Protons through target thick enough to stop all ions - 100 proton histories

Particle	Energy (MeV)	Max Current (micro A)	Flux (ions per second)
p	8-40	60	3.75×10^{14}
d	8-40	30	1.87×10^{14}
$^4He^{2+}$	8-53	30	9.36×10^{13}
$^3He^{2+}$	4-20	60	1.87×10^{14}

Table 6.3: Beam Characteristics of the Scanditronix MC-40

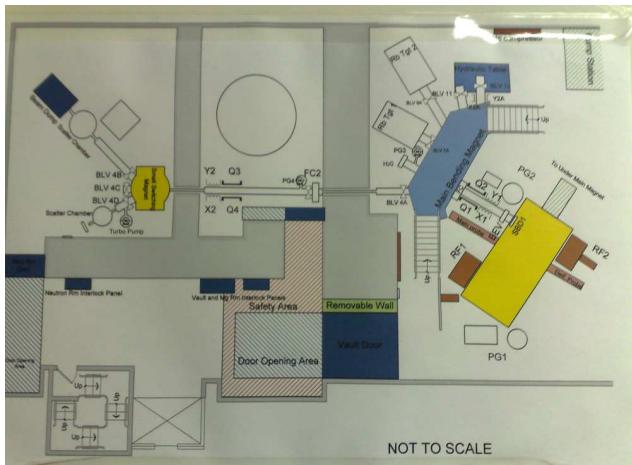


Figure 6.10: Cyclotron layout - main hall and beam lines



Figure 6.11: Gamma warning alarm during 0.5 microamp 36MeV proton irradiation of iron

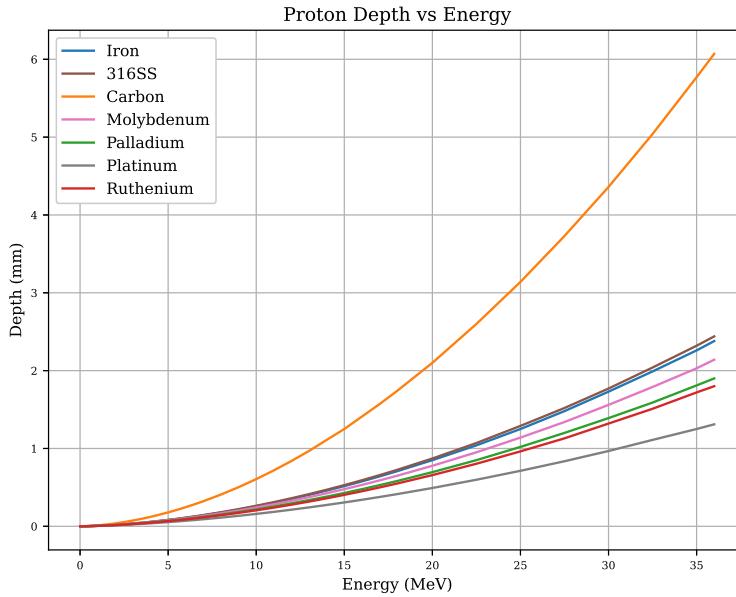


Figure 6.12: Proton stopping distances in various materials

It is clear that 3mm thick samples of Iron or Steel will completely stop 36MeV protons and 1mm of the same is enough to stop any with an energy of 20MeV or less. The target irradiated here, with a thickness of just 0.5mm, will allow most ions to pass through and retain a large proportion of their energy (fig. 6.8b).

6.5.2 Creation of Cobalt-55

$^{55}_{27}Co$ is a relatively short lived isotope with a half life of 17.5 hours and this is a concern to health during and shortly after irradiation. Isotopes with a long half life are not as active in small quantities, and those with very short half life decay away quickly. $^{55}_{27}Co$ decays into another radioactive isotope, $^{55}_{26}Fe$, which then decays into the stable isotope $^{55}_{25}Mn$ (fig. 6.13). The second step in the decay chain is 2.73 years, so the first step with a much shorter half life will be responsible for the majority of the radiation shortly after being irradiated by protons.

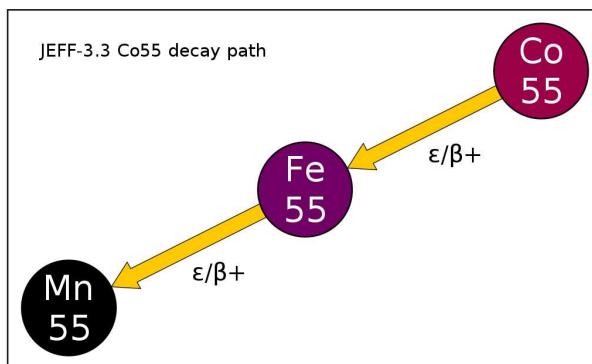


Figure 6.13: Decay path of Cobalt-55

The decay of $^{55}_{27}Co$ will produce a range of gammas (fig. 6.14), but one in particular to focus on when measuring the radioactivity is the 931keV gamma with an intensity of 75%.

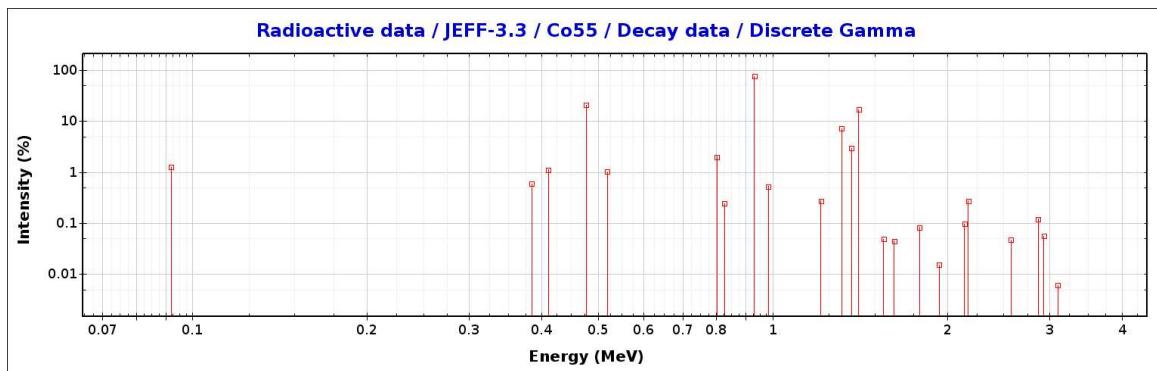


Figure 6.14: Cobalt-55 Gamma Intensity Plot

The stable isotope $^{54}_{26}Fe$ makes up almost 6% of natural iron, and one reaction possibility with a proton results in $^{55}_{27}Co$ and a gamma, $^{54}_{26}Fe(p, \gamma)^{55}_{27}Co$. The most abundant isotope in natural Iron also has a route to transmute to $^{55}_{27}Co$ through the capture of a proton and the loss of two neutrons, $^{56}_{26}Fe(p, 2n)^{55}_{27}Co$. Due to the short half-life of this isotope of Co and the intensity of the 931keV gamma, this peak was expected and measured after irradiating the iron target. It was also estimated manually using the TENDL-2009 and TENDL-2019 cross section databases.

6.5.3 Measurement of Sample Activity

The sample was too radioactive to safely handle immediately after irradiation, so it was left to cool for several days before taking measurements. After it had cooled, a high purity germanium detector (fig. 6.15) was used to measure the activity of the irradiated sample. The detector was calibrated and was expected to detect approximately 4 out of every 100 gammas emitted.



Figure 6.15: High Purity Germanium Detector

The detector and preamplifier are both cooled by liquid nitrogen to 77K, as the band-gap of germanium is 0.7eV and the thermal motion of electrons and nuclei at higher temperatures would induce currents causing noise and interference in the detector. As a gamma ray enters the detector it creates an electron-hole pair in the medium of the detector. The holes and electrons are attracted to the center and to the outer cylinder of the detector, or vice versa.

1 RANGE: 617 = 911.51keV to 649 = 958.79keV

```

2 AREA : Gross = 2127168 Net = 1342565 +/- 2207
3 CENTROID: 631.74 = 933.29keV
4 SHAPE: FWHM = 6.82 FW(1/5)M = 10.06
5 ID: Bi-214 at 934.05keV
6 Corrected Rate = 35349.26 +/- 58.11 cA

```

Listing 6.2: Maestro 931keV Peak Measurement

The gamma counts were recorded over a range of 0MeV to 3MeV (fig. 6.16). Correcting the measured count for the 931keV peak, to account for the geometry and detector, the count rate for 931keV gamma rays from $^{55}_{27}Co$ was measured at $4.43 \times 10^4 \pm 1.05 \times 10^3$ counts per second.

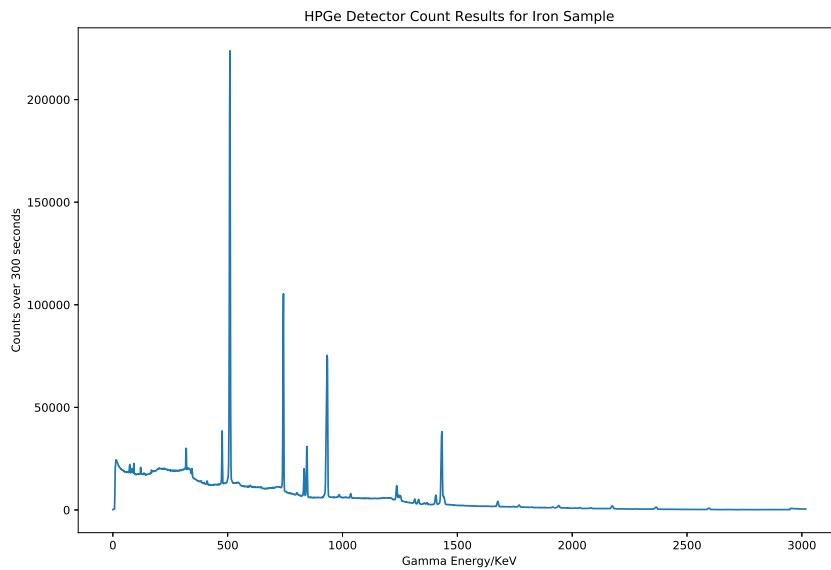


Figure 6.16: Gamma counts from the irradiated iron target over 300 seconds

There are several decay modes that contribute to the radioactivity of the sample, and the largest contributors to gamma emission also undergo either electron capture or β^+ emission. The decay paths are:

- $^{55}_{27}Co \xrightarrow{\epsilon/\beta^+} {}^{55}_{26}Fe \xrightarrow{\epsilon/\beta^+} {}^{55}_{25}Mn$
- $^{56}_{27}Co \xrightarrow{\epsilon/\beta^+} {}^{56}_{26}Fe$
- $^{54}_{25}Mn \xrightarrow{\epsilon/\beta^+} {}^{54}_{24}Cr$
- $^{52}_{25}Mn \xrightarrow{\epsilon/\beta^+} {}^{52}_{24}Cr$
- $^{52}_{25}Mn^* \xrightarrow{\epsilon/\beta^+} {}^{52}_{24}Cr$

Where positrons are emitted, they will annihilate with an electron. This event will release two photons in opposite directions, each of which will have a 511keV energy. This is responsible for the 511keV peak measured in fig. 6.16. However, at the time of writing, the computer code will not compute this peak and it will be omitted from the computational results.

6.5.4 Prediction of Activity

Following the measurement of the thin target, in the first instance the predicted activity will be estimated using the average ion energy, the cross section of this energy and the two reactions that result in Co-55 (${}^{54}_{26}Fe(p, \gamma) {}^{55}_{27}Co$)

and $^{56}_{26}Fe(p, 2n)^{55}_{27}Co$. The activity code (version 1 and version 2) are then used to predict the activity of the sample as well as the expected gamma spectra.

6.5.5 Estimation of Cobalt-55 Radioactivity

There are over ten TENDL libraries available, and the available data and the range of data points is inconsistent across these different versions. The earlier library, that was used in the first version of the Activity code, covers a range up to 200MeV for Iron-proton reaction cross sections. Many of the more recent data files contain total reaction cross sections only, and in the most recent data files, 2017 and 2019, the energy range only covers proton reactions from 0MeV to 28MeV, abruptly dropping to 0 barns at 30MeV as seen in fig. 6.17.

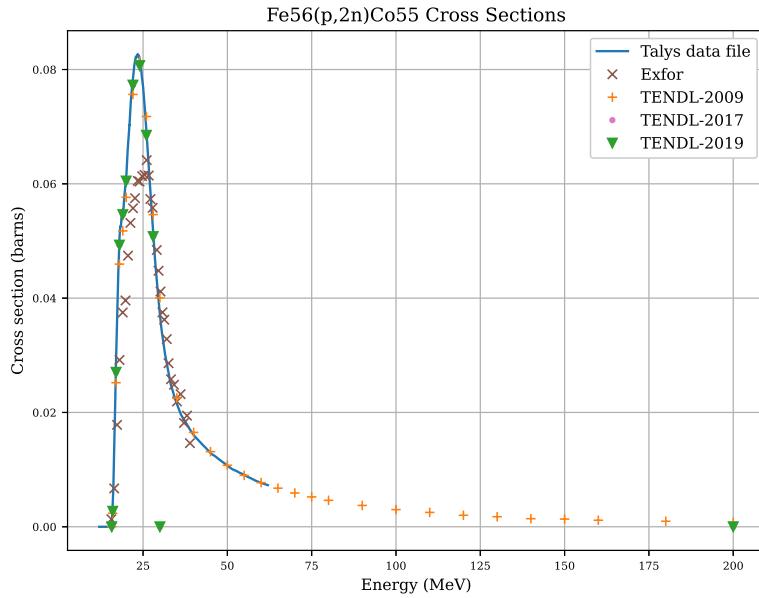


Figure 6.17: $Fe56(p, 2n)^{55}Co$ cross section data from several sources [104][196][197][198][199]

The activity of the sample, in particular due to the radioactivity of ^{55}Co , was estimated using a number of data sources. As a consequence of the short beam duration when compared to the half-life of Cobalt-55, any loss during irradiation is ignored in each case.

The estimations use one set of experimental data, from the EXFOR data file, as well as evaluated and calculated data points. As may be seen in 6.17, all data sets follow a similar path although the peak is slightly lower for the experimental data. The two most recent TENDL data plots terminate abruptly, but the TENDL-2009 and TALYS calculated data points continue in agreement with one another up to 62MeV, where the calculated points end.

The parameters in relation to the beam current, duration, target material and the gamma intensity were consistent between the estimations. They are the same parameters as used in the experimental work and are given in table 6.4.

The 2019 data file ranges up to 30MeV for the $^{54}_{26}Fe(p, \gamma)^{55}_{27}Co$ and $^{56}_{26}Fe(p, 2n)^{55}_{27}Co$ reactions. As the data cuts off at 30MeV, the cross section data for a 28-30MeV proton is used in the estimation. This results in an over estimate in activity.

A 36MeV proton is expected to lose up to 4-5MeV travelling through a 0.5mm thick Iron target, as seen in fig. 6.8c. However, in this estimation the ions are assumed to be only 36MeV (or 28MeV, where 36MeV data are unavailable).

Parameter	Value
ND/atoms $^{56}_{26}Fe$	7.4×10^{28}
ND/atoms $^{54}_{26}Fe$	4.6×10^{27}
J/amps	0.5×10^{-7}
Q/C	1.60×10^{-19}
d/m	5.0×10^{-4}
irradiation time/s	3.0×10^2
Co55 t-half/s	6.3×10^4
Co55 λ	1.1×10^{-5}
931keV Gamma Intensity	0.75

Table 6.4: Parameters used to estimate Co55 931keV gamma activity

	TENDL 2009	TENDL 2017	TENDL 2019	Exfor	TALYS file	TALYS DB
$^{54}_{26}Fe(p, \gamma)^{55}_{27}Co$						
P energy (MeV)	36	28	28	36	36	36
XS σ (barns)	1.29×10^{-4}	1.94×10^{-4}	1.94×10^{-4}	-	1.33×10^{-4}	1.75×10^{-4}
Reaction Rate	9.32×10^4	1.40×10^5	1.40×10^5	-	9.64×10^4	1.27×10^5
Activity (Bq)	1.78×10^1	2.69×10^1	2.69×10^1	-	1.84×10^1	2.43×10^1
Gamma (Bq)	1.34×10^1	2.02×10^1	2.02×10^1	-	1.38×10^1	1.82×10^1
$^{56}_{26}Fe(p, 2n)^{55}_{27}Co$						
P energy (MeV)	36	28	28	36	36	36
XS σ (barns)	2.13×10^{-2}	5.08×10^{-2}	5.08×10^{-2}	2.30×10^{-2}	2.00×10^{-2}	2.00×10^{-2}
Reaction Rate	2.44×10^8	5.81×10^8	5.81×10^8	2.64×10^8	2.29×10^8	2.29×10^8
Activity (Bq)	4.67×10^4	1.11×10^5	1.11×10^5	5.04×10^4	4.38×10^4	4.38×10^4
Gamma (Bq)	3.50×10^4	8.33×10^4	8.33×10^4	3.78×10^4	3.29×10^4	3.29×10^4

Table 6.5: Estimation of the Cobalt-55 931keV gamma activity based on the TENDL-2009[104], TENDL-2017[196], TENDL-2019[197] data files, EXFOR experimental data [198] and TALYS generated data files[199].

The largest contribution to the 931keV gamma is due to the $^{56}_{26}Fe(p, 2n)^{55}_{27}Co$ reaction, with the $^{54}_{26}Fe(p, \gamma)^{55}_{27}Co$ reaction being responsible for less than 0.1% of the gammas.

Out of the estimates calculated, the outliers are those that used data files with a 28 MeV maximum limit, and this is as to be expected. The final three results are much closer to one another, ranging from 43,800Bq to 50,400Bq for the overall activity and 32,900Bq to 37,800Bq for the rate of emission of Gammas. The full details including cross sections and reaction rates used are given in table 6.5.

6.5.6 Simulating Proton Irradiated Iron with Activity V1

The first version of the Activity code with modified Bateman equations was used to calculate the predicted radioactivity (fig. 6.18) and predicted gamma lines (fig. 6.19). A full output file and details of all isotopes and their activities are given in appendix D.

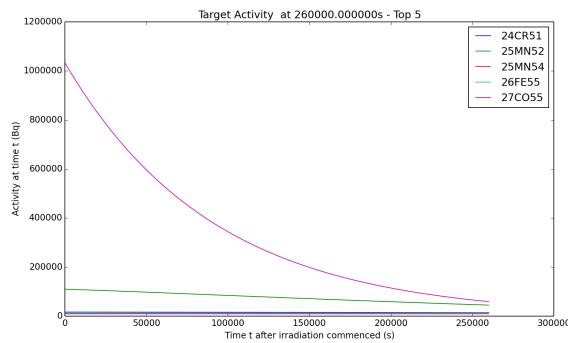


Figure 6.18: Isotopes with the top 5 radioactivity plotted over approximately 3 days

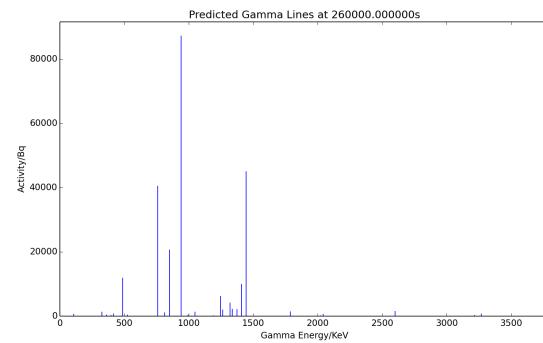


Figure 6.19: Predicted gamma lines approximately 3 days after irradiation

The code predicts a final 931keV gamma activity of over 59,000 Bq and highlights several other isotopes of interest, including $^{52}_{25}Mn$, $^{51}_{24}Cr$, $^{55}_{26}Fe$ and $^{54}_{25}Mn$, all of which have an activity of approximately 10,000 Bq or above. After 3 days of cooling, the target is predicted to dose an 80KG human, 1 meter away from the target, with 4.0×10^{-11} Gy/s (absorbed dose) which is less than the limit for a member of the public (6.34×10^{-10} Si/s, effective dose).

6.5.7 Simulating Proton Irradiated Iron with Actvity V2

The iron irradiation experiment was modelled using the second version of the Activity code. In this version the entire cross section database was created using the TALYS computer code (6.3.3). The same decay equations used in version 1 were also used here.

As the iron sample is irradiated a large number of particles (neutrons, protons, deuterons, tritons, helium-3, alpha particles and gamma rays) are created. Whilst the beam is on, these are produced at a rate of more than 10^9 per second (figure 6.20).

As irradiation commences, the isotope of iron responsible for most of the non-particle radiation is $^{54}_{26}Fe$, despite having a natural abundance of only 5.8%. Two residuals in particular resulting from proton reactions with $^{54}_{26}Fe$ are responsible for the activity and these are $^{54}_{25}Co$ and $^{53}_{26}Fe$.

With a short half life of 0.19 seconds, $^{54}_{25}Co$ is responsible for the majority of decays per second within a few seconds of irradiation starting. However, at this same time the majority of the gamma energy from residual isotopes originates from the decays of $^{53}_{26}Fe$ (meta 1) and $^{52}_{25}Mn$ (meta 1). The average gamma energy per decay of these isotopes is 3.03MeV and 1.42MeV respectively, and this is the reason for such a high contribution towards the gamma energy output (6.21a).

By the time the beam is turned off, at 300s, the primary isotope contributing almost 50% of the decays per second is $^{53}_{26}Fe$. Its contribution to the overall gamma output is still quite low with the main contributor being $^{52}_{25}Mn$ (meta 1) as was the case at the start of irradiation (6.21b). This difference between the decays and gamma energy is due to $^{53}_{26}Fe$ having an average gamma energy of only 0.18MeV per decay, whereas the metastable $^{52}_{25}Mn$ has a much higher average gamma energy per decay of 1.4MeV.

Once the beam is turned off, those isotopes with a very short half life decay away quickly. After 1 day of cooling, the isotopes responsible for activity in the sample are very different. With a half life of less than 10 minutes, the $^{53}_{26}Fe$ will have decayed to nothing as would have the $^{53}_{26}Fe$ (meta 1) and $^{52}_{25}Mn$ (meta 1). At this time, the isotope with the highest contribution to decays per second will be $^{55}_{27}Co$, but with a half-life of 17.53 hours it does not remain the largest contributor. It is also responsible for more than half of the gamma power output at this time as well (6.22a).

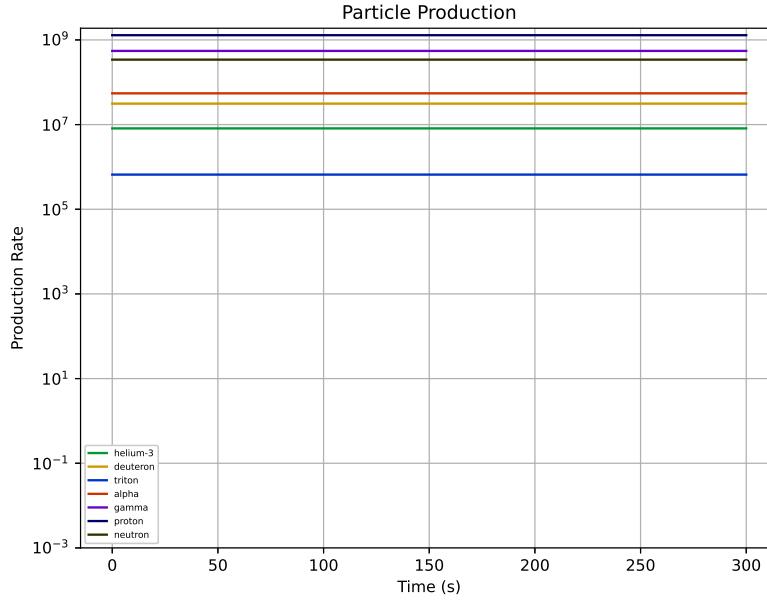


Figure 6.20: Particle production rate in beam

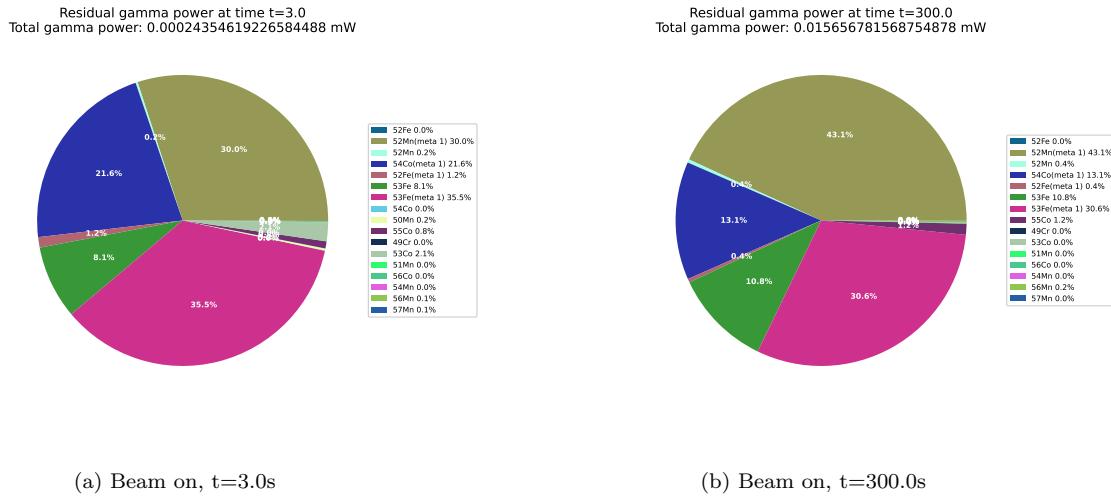


Figure 6.21: Gamma output of residual isotopes, by isotope, at the start and end of irradiation

At the end of the experiment, the time at which the experimental gamma readings were taken (2.6×10^5 s), the largest contributor to both decays per second and gamma energy output per second is the $^{52}_{25}Mn$ isotope (6.22b).

The computer code also calculates the predicted gamma lines as time goes by. The gamma lines become more prominent whilst the ion beam is irradiating the target to a maximum activity at 300 seconds. The three most active gammas at this point are computed to be 1.43MeV, 378keV and 701keV. The 1.43MeV gamma is released by the decay of the metastable isotope $^{52}_{25}Mn$ (meta 1) and is emitted at a rate of 2.9×10^7 Bq. The 377keV and 791keV gammas are emitted by $^{53}_{26}Fe$ and $^{53}_{26}Fe$ (meta 1) respectively having emission rates of 2.4×10^7 Bq and 9.7×10^6 Bq.

Following a day of cooling the 931keV gamma from $^{55}_{27}Co$ will have an activity of approximately 2.9×10^5 Bq and this is accompanied by another gamma of similar energy. This is the 935keV gamma from the decay of $^{52}_{25}Mn$. At this point in time the activity of the 935keV gamma is less than that of the 931keV gamma with an activity of 9.49×10^4 Bq. In addition, the $^{52}_{25}Mn$ will also emit a 1.43 MeV gamma at a similar rate calculated

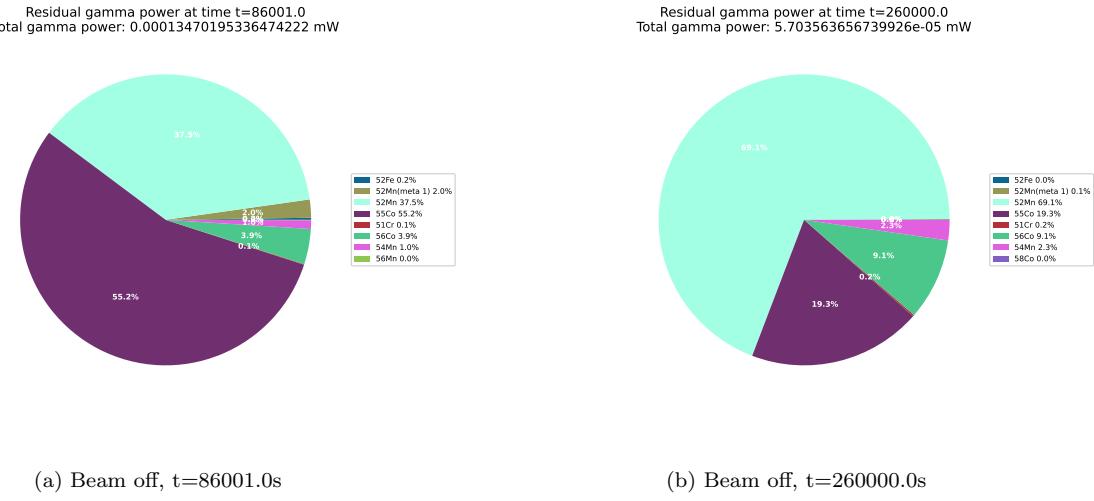


Figure 6.22: Residual isotope power output, by residual, once beam is off.

to be $1.0 \times 10^5 \text{Bq}$. With half lives of 17.5 hours for ^{55}Co and 134 hours for ^{52}Mn it is clear that, at some point, the activity of the 931keV gamma will drop below that of the 935keV gamma.

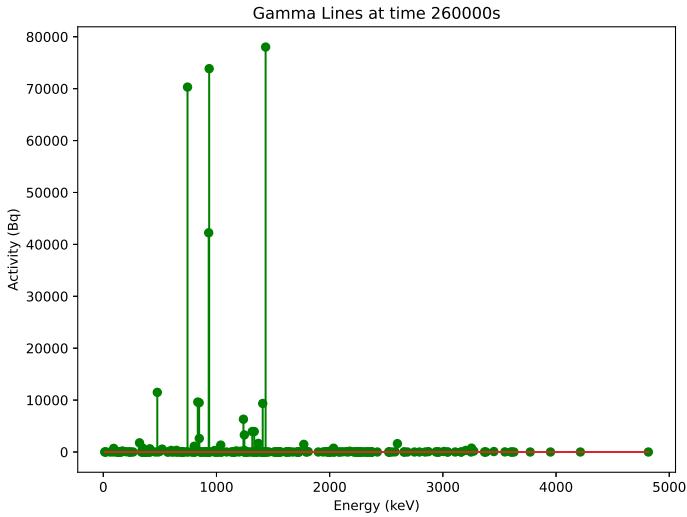


Figure 6.23: Expected gamma lines by the end of the experiment (3 days)

At the end time of the simulated experiment, $2.6 \times 10^5 \text{s}$, the activity of the 931keV Gamma is predicted to have dropped to just over 42,000Bq. It's neighbour, the 935keV Gamma, will have an activity almost double, with 74,000 decays per second (fig. 6.23). The full results calculated by this code are included in appendix R.

The prediction of such a large neighbouring Gamma does raise a question where the experimental value is concerned. It may be that, due to the resolution of the Gamma detector, the measured peak is the sum of both the 931keV and 935keV Gamma. Ideally the detector would be calibrated for a smaller range about that peak and the experiment would be repeated to determine the value of both the 931keV and 935keV peak.

6.5.8 Comparing Experimental Results with Calculated Values

The Co-55 gamma ray value was measured in section 6.5.3. A crude estimation using cross section data files was carried out in section 6.5.5 and this was followed by the use of both versions of the Activity code (section 6.5.6 and section 6.5.7).

The estimation, using the cross section data files only (TENDL, EXFOR), is cruder and does not take into account the details of the energy lost by the projectile as it passes through the target. It also does not take into account the creation of ^{55}Co through other reactions. It gives a lower value of activity than the computer codes and the experimentally measured value (table 6.6).

Value source	Co-55 Gammas/second at approximately 3 days
Experimental	$4.4 \times 10^4 \pm 1.05 \times 10^3$
TENDL-2009	3.5×10^4
Activity V1	5.9×10^4
Activity V2	4.2×10^4

Table 6.6: Comparison of proton irradiated iron Co55 931keV gamma rates

Although the predictions are in good agreement with the experimental value, there are still a number of questions that remain unanswered. The most important of these is whether or not the 935keV peak has been included with the 931keV peak when the measurement was taken.

6.6 Predicted Activity for 100DPA Irradiated Iron

Generation IV reactors are required to withstand 100-150DPA over the lifetime of the plant, which may span 30 years or more. The Scanditronix cyclotron is capable of creating a 60 microamp beam and, due to the small area (and volume of material) this may be concentrated on, it is able to create high damage rates over a shorter period of time.

The Activity V2 code was used to calculate how radioactive a 100DPA iron sample would be after irradiation at the maximum current setting of 60 microamps. The simulated target is 0.5mm thick, pure iron and the beam is protons and has an area of 64mm^2 . The projectile fluence is 3.75×10^{14} protons per second, and the number of atoms within the volume of iron are approximately 2.56×10^{21} . Five different energy settings were used (5MeV, 10MeV, 15MeV, 20MeV, 25MeV) and the vacancies per ion as well as the ion trajectories were generated by SRIM.

	5MeV	10MeV	15MeV	20MeV	25MeV	30MeV
VPI	38.5	58.8	51.0	23.7	20.2	12.8
DPA/year	178	272	236	109	93	59
Days for 100 DPA	205	134	155	333	391	617
Damage Depth (mm)	<0.1	0.25	0.5+	0.5+	0.5+	0.5+
Activity ¹ (Bq)	2.27×10^6	7.32×10^8	2.55×10^9	3.66×10^9	4.10×10^9	4.88×10^9
Gamma Output ¹ (mW)	9.44×10^{-5}	3.87×10^{-1}	1.32×10^0	1.29×10^0	6.05×10^{-1}	5.74×10^{-1}
Activity ² (Bq)	2.14×10^6	6.86×10^8	2.35×10^9	3.08×10^9	2.82×10^9	3.52×10^9
Gamma Output ² (mW)	8.85×10^{-5}	3.64×10^{-1}	1.24×10^0	1.16×10^0	3.63×10^{-1}	2.94×10^{-1}

Table 6.7: Comparison of proton irradiated iron up to 100DPA at a range of proton energies. The first set of measurements¹ are calculated values at the point the proton beam is turned off. The second set² are calculated after 1 week of cooling.

In each case the sample was irradiated close to saturation. Doubling the irradiation time would not be expected to significantly increase the activity, as sufficient time (in each case) is given for the radioactive isotopes being created to balance those decaying away.

It is clear to see from the predicted activities in table 6.7 that the higher energy beams would take longer to reach 100DPA, as they pass through the target leaving at a high energy, whilst also irradiating the target several orders of magnitude more than 5MeV and 10MeV.

The Coulomb potential between a proton and an Fe nucleus, at the combined nuclear radius of both, is approximately 6.5MeV (eq. 4.2 and eq. 4.3). Unlike neutron radiation, protons need to overcome the Coulomb repulsion and this explains the relatively low radioactive activation by the 5MeV beam.

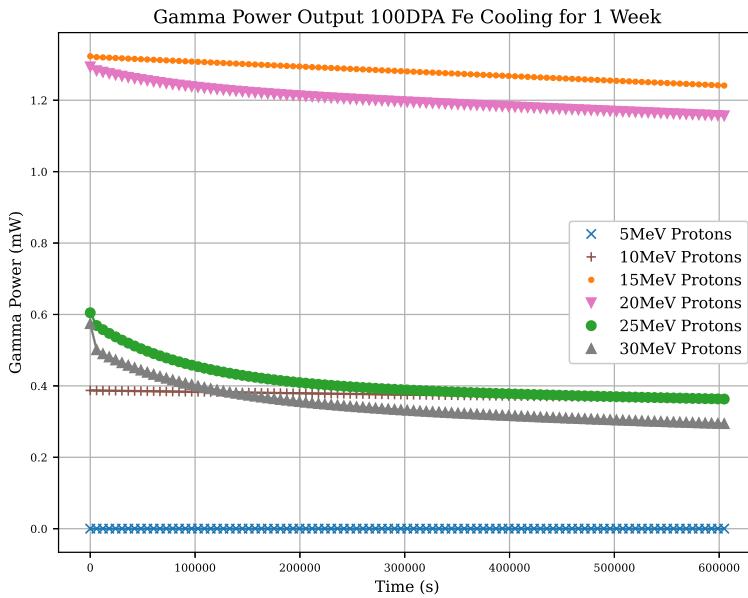


Figure 6.24: Change in isotope gamma output over time with proton beam off.

The predominant isotope responsible for radiation in the 5MeV sample is expected to be $^{58}_{27}Co$, and with a half life of 77 days this will remain radioactive for a substantial amount of time (relative to testing by irradiating, cooling then testing the material).

For the 10-20MeV irradiated samples the primary isotope responsible for the radiation is $^{56}_{27}Co$, and its contribution to the total gamma output in terms of power ranges from 95% to 99%. This isotope has a similarly long half life, relative to the testing cycle, of 71 days. Depending upon the volume of material being irradiated, this could lead to problems whereby the sample may need to cool for months to reach a safe activity to handle.

As higher energy protons are used to damage the sample a wider range of reaction possibilities become more likely including that leading to both $^{52}_{25}Mn$ and $^{52}_{25}Mn$ (meta 1). By the end of irradiation a large percentage of radioactivity is due to these isotopes (28.6% for 25MeV protons and 42.8% for 30MeV protons). The half life of $^{52}_{25}Mn$ and $^{52}_{25}Mn$ (meta 1) are short, being 5.6 days and 21 minutes respectively, and will decay away leaving $^{56}_{27}Co$ as the main contributor to radiation once again.

The overall gamma ray activity during cooling, for each proton energy, is plotted in fig. 6.24 and this shows the least radioactive at the end of cooling to be 5MeV protons, followed by 30MeV protons. Detailed plots resulting from this Activity V2 simulation are in appendix E.1.

6.7 Proton Irradiated Molybdenum

A colleague at the University of Birmingham, John Hewett, irradiated several samples of Molybdenum for materials testing and throughout this the activity of the Molybdenum was measured by the department's High Purity Germanium (HPGe) detector[200]. Similar to the Iron sample, thin sheets of Molybdenum were irradiated using a beam line from the cyclotron.

6.7.1 Molybdenum Experimental Irradiation Results

A thin 49 micrometer sheet of pure Molybdenum was irradiated with a 13MeV proton beam for 19 seconds at a current of 0.5 microamps. The beam area on the target was 50mm^2 . A thick 506 micrometer sheet of the same was also irradiated, but this time a higher beam current of 5 microamps and a duration of 1,500s.

The detector was calibrated to detect energies in the range covering 0MeV to 1.2MeV and several measurements were taken for each sample. For the thick target the total gamma activity was measured as 16MBq at 3 days (14MBq for gammas above 200keV) and 7.1MBq at 7 days (6.6MBq for gammas greater than 200keV).

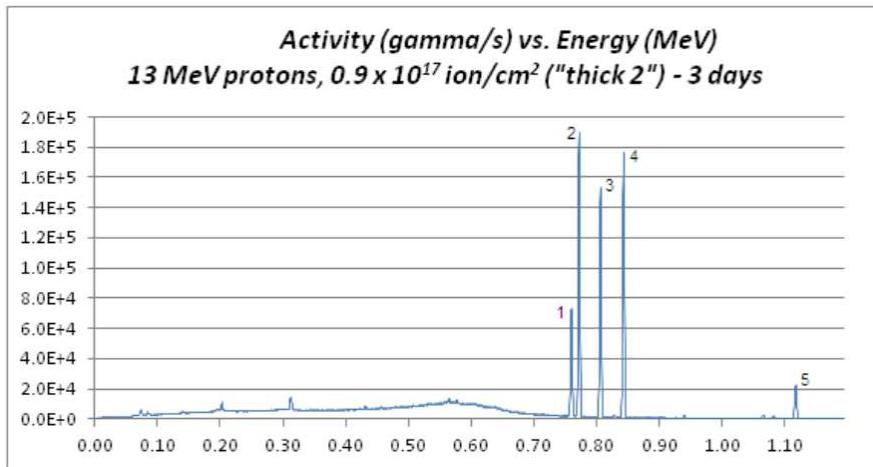


Figure 6.25: Proton irradiated Molybdenum gamma activity measured by colleague John Hewett

A full gamma spectra was also recorded for the thick sample by gamma peak at three days. The isotopes responsible for the five most prominent peaks at this point were $^{95}_{43}\text{Tc}$ (half life of 20 hours[26]) and $^{96}_{43}\text{Tc}$ (half life of 4.28 days[26]). The results measured and interpolated by my colleague are given in fig. 6.25 and table 6.8.

Peak	Energy (keV)	Isotope	Gamma Intensity	Activity (Bq)	Measured Energy (keV)
1	766	$^{95}_{43}\text{Tc}$	93.82%	5.118×10^5	763
2	778	$^{96}_{43}\text{Tc}$	99.76%	1.355×10^6	776
3	813	$^{96}_{43}\text{Tc}$	82.00%	1.155×10^6	810
4	850	$^{96}_{43}\text{Tc}$	97.57%	1.386×10^6	847
5	1127	$^{96}_{43}\text{Tc}$	15.16%	2.103×10^5	1123

Table 6.8: Gamma emissions measured from a 0.5mm thick sample of Molybdenum irradiated with 13MeV protons at 5 microamps for 1,500 seconds

6.7.2 Estimating the Activity of Proton Irradiated Molybdenum

The gamma activity of the 0.5mm thick molybdenum target will be estimated using the generated TALYS cross section files. The generated data are compared to a number of experimental data sources through EXFOR as well as the TENDL-2019 data file. In this estimate, there is no ion transport, but the average cross section reaction across the energy range is computed using the TALYS generated data. This is then used to compute the reaction rate. Simple decay equations are then used to compute the activity of the residual radioactive isotopes.

Naturally occurring Molybdenum is made up of four stable isotopes, two observationally stable isotopes, $^{92}_{42}\text{Mo}$ and $^{98}_{42}\text{Mo}$ and one unstable isotope, $^{100}_{42}\text{Mo}$, with an extremely long half life of 9.9×10^{19} years (fig. 6.26).

Molybdenum Natural Abundance of Stable Isotopes

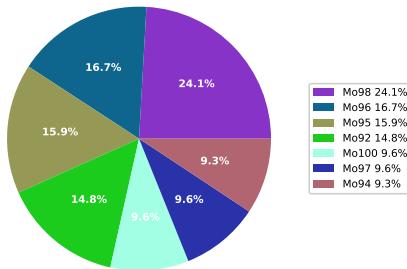


Figure 6.26: Isotopes in naturally occurring Molybdenum

There are five reactions of particular interest that are predicted to create radioactive isotopes at the highest rate. Two of the residual isotopes were identified in the experimental work and the reactions responsible are the first two of the five listed below.

- $^{95}_{42}Mo(p, n)^{95}_{43}Tc$
- $^{96}_{42}Mo(p, n)^{96}_{43}Tc$
- $^{96}_{42}Mo(p, n)^{96}_{43}Tc$ (meta 1)
- $^{97}_{42}Mo(p, n)^{97}_{43}Tc$
- $^{94}_{42}Mo(p, n)^{94}_{43}Tc$

The isotope $^{95}_{43}Tc$ produced four measurable gamma activities, and these gamma decay rates will be estimated using cross section data available from TENDL and EXFOR. The cross sections have been measured experimentally, with their results published, by four groups since the 1970s. Whilst there are discrepancies, the total cross section predicted by the TALYS code fits well with the TENDL-2019 database and measurements by Lamere[201] and Flynn[202]. The lower energy cross sections of Hogan[203] do not match as well but the higher energy reactions(30MeV+) are a closer match. Finally cross section measurements by Levkovski[204] are slightly larger near to the peak values, but on the whole match well with the TALYS computed values.

The cross sections computed by the TALYS code are split into both the reaction to the Technetium-96 ground state and metastable 1 state. The separate data are plotted in fig. 6.27 along with their combined cross section.

This estimate will calculate the contribution from the ground state only. The metastable state of $^{96}_{43}Tc$ has a 2% chance to decay directly to $^{96}_{42}Mo$ and a 98% chance to decay to the ground state. The half life for the metastable state is 51.5 minutes[26] and, coupled with a lower reaction cross section than the ground state, it will contribute a relatively small amount of $^{96}_{43}Tc$ when compared to those created by the ion beam and will delay the production of the ground state past the point at which the beam is removed.

The 13MeV protons will lose all their energy within the first 375 micrometers of the material as computed by SRIM (fig. 6.12). As the ions lose energy the probability of a reaction occurring changes. The cross section ranges from a maximum of 475mb at 11MeV and this drops to zero once the ion energy falls to 3.8MeV. The average cross section over the entire energy range based on SRIM stopping distance calculations and TALYS data is 0.269 barns.

Given the data used is in good agreement with EXFOR experimental data, the 13MeV is expected to transmute 2.93×10^9 $^{96}_{42}Mo$ atoms into $^{96}_{43}Tc$ every second. By the end of the beam almost 4.4×10^{12} atoms will have been transmuted via this reaction and, as the half life of $^{96}_{43}Tc$ is 247 times greater than the 1,500s beam duration,

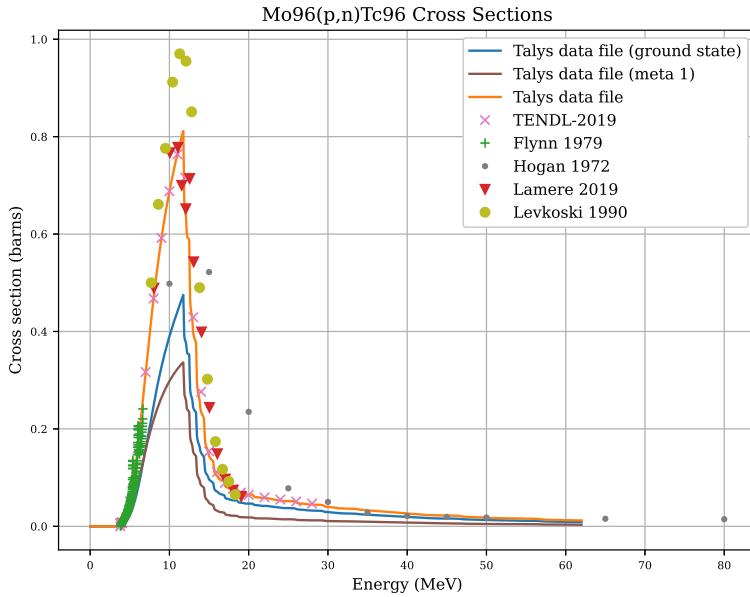


Figure 6.27: Reaction cross sections for $^{96}_{42}\text{Mo}(p, n)^{96}_{43}\text{Tc}$ from various sources

atoms lost to decay during irradiation will be ignored in this estimate. The intensity of each gamma varies, and these are listed in 6.8.

Peak	Energy (keV)	Measured Activity (Bq)	Estimated Activity (Bq)	Factor of Over Estimate
2	778	1.355×10^6	4.74×10^6	3.5
3	813	1.155×10^6	4.15×10^6	3.6
4	850	1.386×10^6	4.93×10^6	3.6
5	1127	2.103×10^5	7.66×10^5	3.6

Table 6.9: A comparison between measured and estimated peaks from a 0.5mm thick sample of Mo irradiated with 13MeV protons at 5 microamps for 1,500 seconds

There is a discrepancy between the measured amount and that predicted by past work, and it is consistent across the four gamma peaks. There are several simplifications with this estimate including the omission of the metastable state, but the inclusion of this would increase the estimated activity rather than decrease it.

My colleague investigated the affect of self attenuation of gammas passing through the Mo sample. They made the assumption that the radioactive isotopes were concentrated half way through the thickness of the sample and that the gammas would need to pass through 250 micrometers of matter. It was expected that gammas above 200keV would retain 94% of their intensity with those above 1MeV having 98.5% of their original intensity[200].

The distribution of radioactive isotopes would in fact be closer to the surface of the material than the midway point. This is illustrated in fig. 6.28 and it shows the majority of the $^{96}_{43}\text{Tc}$ is created in the top 150 microns. This plot is based on the SRIM stopping range for protons in Mo and the cross section data used in the estimate.

It is plausible that the orientation of the sample would affect the gamma count. If it were placed such that the beam irradiated side faced the detector there would be less material between the radiation source and the detector. The opposite way round and more material would block the path of the gammas to the detector, thus decreasing their intensity.

When plotting the gamma attenuation through Mo over a range of gamma energies it is clear that those with an energy of 500keV or more drop to an intensity of as low as 0.9 in targets up to 1mm thick (fig. 6.29)[39]. This is double the target depth of the thick target and so it does not explain the discrepancy between the

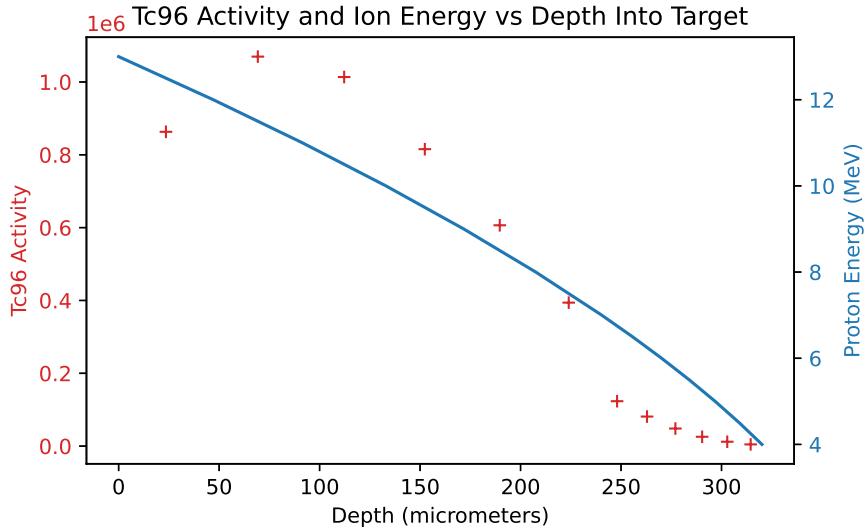


Figure 6.28: Tc activity and beam energy as a function of depth into the target, showing a non-uniform distribution of radioactive atoms in the sample

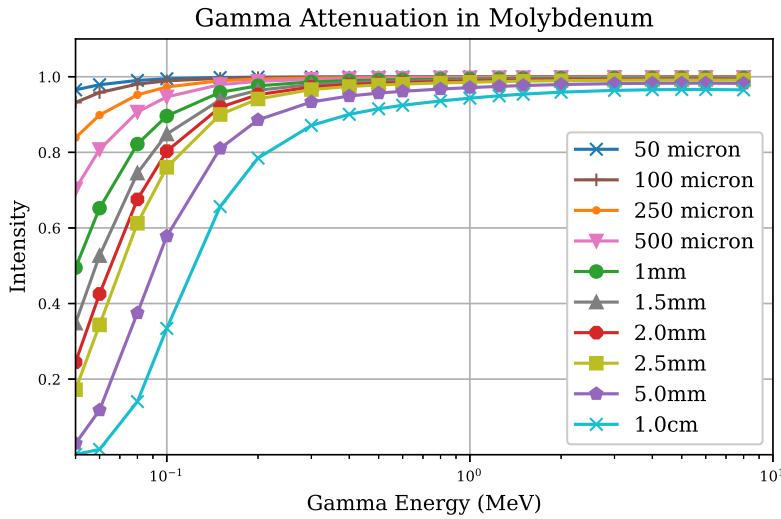


Figure 6.29: The attenuation of Gammas through Mo for a range of energies and thicknesses[39]

experimental and estimated values.

6.7.3 Simulated Proton Irradiated Molybdenum

Both the thin and thick samples irradiated experimentally are simulated. However, detailed data are available for the thick sample only. A qualitative comparison of the thin sample will be made with the experimental results available.

Only stable isotopes are included in the cross section data file currently and, as 3 out of the 7 “stable” isotopes are either observationally stable, or have an incredibly long half life, their reactions are not included. The addition of these isotopes could highlight additional reactions but the data would need to be created first, if possible, using TALYS.

Out of the four stable isotopes of Mo, ^{96}Mo is responsible for most of the radioactivity both at the time the beam is turned off (1,500s) (fig. E.27) and after 3 days of cooling (fig. E.28). As discussed earlier this is due to the creation of both the ground state and metastable 1 state for ^{96}Tc . Whilst the beam is on over 60%

of the activity is due to the metastable-1 isotope. The gamma energy emitted by the isomeric transition is comparatively low with an energy of just 32.28keV, and as a result it's contribution to the power output is also low. In contrast the transition to the ground state for $^{96}_{43}Tc$ (meta 1) releases many energetic gammas. The majority of the energy released towards the end of the ion irradiation is due to both the ground state and metastable 1 of $^{94}_{43}Tc$ (fig. E.31).

After three days of cooling almost 67% of the decays are due to the ground state $^{96}_{43}Tc$ isotopes in the sample with another 31% from $^{95}_{43}Tc$ (fig. E.30). The gamma energy output is also primarily due to these two isotopes, although with an 87% and 12% split (fig. E.32).

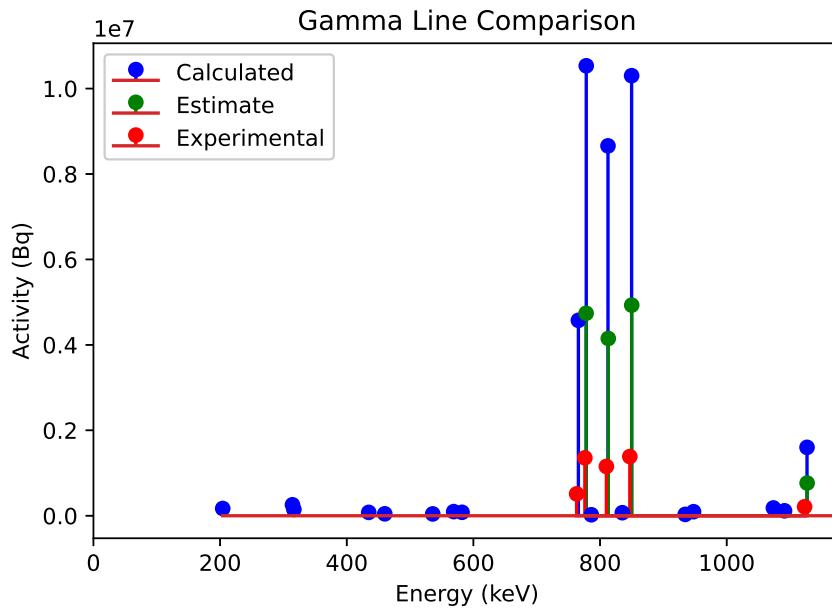


Figure 6.30: Comparison of Gamma peaks from experiment, a simple estimate and the Activity code.

It is clear that there is a disagreement in the values of the experimental, estimated and computed activities and gamma lines (fig. 6.30). The fact that the estimated activities are lower than the calculated values was expected as the estimate only considered one reaction. The difference between the experimental work with both the estimated and calculated values is harder to explain but will be discussed in section 8.2.4. The full results from the activity code are included in appendix E.2.

Peak	Energy (keV)	Measured Activity (Bq)	Computed Activity (Bq)	Factor of Over Estimate
1	766	5.118×10^5	4.574×10^6	8.9
2	778	1.355×10^6	1.053×10^7	7.8
3	813	1.155×10^6	8.658×10^6	7.5
4	850	1.386×10^6	1.030×10^7	7.4
5	1127	2.103×10^5	1.601×10^6	7.6

Table 6.10: A comparison between measured and calculated peaks from a 0.5mm thick sample of Mo irradiated with 13MeV protons at 5 microamps for 1,500 seconds

6.8 Proton Irradiated Molybdenum Doped Steel

Another colleague in the Metallurgy and Materials department at the University of Birmingham, Alex Dickinson-Lomas, requested a copy of the Activity code and a simulation of a steel with a specific composition (Fe: 96.375, C: 0.772, Cu: 0.024, Mn: 1.36, Ni: 0.698, Si: 0.381, Cr: 0.092, V: 0.008, P: 0.009, S: 0.003, Mo: 0.278) that would be irradiated with 5MeV protons.

In comparison to the previous cases, 36MeV protons in Fe and 13MeV protons in Mo, the activity is much lower. The simulated beam current was 0.5 microamps and the duration 5 minutes, but in contrast to the 100MBq maximum activity of the Iron sample, this is expected to reach a maximum of just over 10KBq with a gamma output of just 1.0×10^{-8} mW.

During irradiation, the isotope responsible for the majority of the residual radioactivity is $^{13}_7N$ and this is created through the $^{13}_6C(p,n)^{13}_7N$ reaction. This is despite the stable isotope carbon-13 having a percentage by mass in the sample of 0.008322%.

The reaction probability for $^{13}_6C(p,n)^{13}_7N$ is also high in the 4-5MeV range with a cross section between 0.1 and 0.15 barns. In contrast the first reaction of note for Carbon-12 results in Boron-9 and this begins at proton energies near to and above 10MeV.

After a day of cooling the activity due to Nitrogen-13 will have decayed through 144 half lives to nothing. The reaction responsible for almost 40% the radioactivity at this point is predicted to be $^{58}_{26}Fe(p,n)^{58}_{27}Co(meta1)$. Other residual isotopes of note are $^{57}_{27}Co$, $^{58}_{27}Co$, $^{55}_{26}Fe$ and $^{55}_{27}Co$. A full breakdown is included in appendix (E.3).

Whilst there are no experimental activity data to compare to, the Activity code is a useful tool. It takes a complex material and highlights likely reaction interactions, highly radioactive residual isotopes and gamma peaks. It also predicts the total activity expected and the calculation may be run multiple times with a range of target thicknesses and beam energies.

6.9 Media Accessible via GitHub

The activity code outputs several videos to show how the activity and gamma output of a target changes over time, during and after irradiation. Several of these are available through the wiki page for the code at the URL given below.

<https://github.com/BenPalmer1983/activityppy/wiki>

6.10 Concluding Remarks on the Activity Code

Existing isotope inventory codes exist, but many are either designed to calculate decay chains, only considering decay rates and branching, or for neutron activation. Methods exist to couple neutron transport codes with these inventory codes, for example FATI, MCNP and FISPACT. Other Bateman solvers will exist, but they may be proprietary and will offer binaries only with no access to the source code.

The Activity code is designed specifically for activation with light ions. Decay rates, branching factors and source rates are included in the Bateman solver, so there is no need to make the assumption that saturation is reached for all radioactive isotopes. It is open source and the code may be examined and modified by anyone. It is also designed to highlight problematic isotopes during and after irradiation.

Chapter 7

Potentials for Iron, Palladium and Ruthenium

The aim of this section is to derive an interatomic potential for Fe-Pd and Fe-Ru that are suitable for modelling both radiation damage and a change in concentration of PGMs at the grain boundary. Two computer codes, QECONVERGE and QEOS, were written to automate parameter convergence for PWscf and to calculate bulk properties. These data join many other DFT calculations to form a reference database for fitting the potentials. A fitting code EAMPA was developed and used to derived the two binary alloy potentials.

7.1 Introduction

7.1.1 Motivation for Derivation of Potentials

Potentials exist for the pure elements BCC Fe, FCC Pd, HCP Ru and binary alloy Fe-Pd. The literature has shown that there is a need to craft potentials to reproduce desired features, such as bulk properties[43] or stacking fault energies[205]. Transferability of potentials is also a known problem[206] and those created for a particular model do not necessarily work when the model is changed. Existing potentials for BCC Fe and FCC Al are used to compute the properties for FCC Fe and BCC Al in this work by comparing with DFT computed values in order to highlight this issue.

The potentials developed here are fit to DFT data for FCC Fe, due to the steel being austenitic, marking the first departure from existing potentials. The intention is to create potentials to allow either MD or a combination of MD with akMC to model radiation damage and how this affects the material at grain boundaries over long periods of time. The potentials for each element require the capability to model the damage process, and so each has a ZBL potential that dominates over the embedding energy at small atom separations.

Due to the damage the potentials are expected to model, the potentials were also fit to DFT computed forces and energies as a result of both defects and lattices with atoms perturbed from their perfect locations. Finally, there was an effort to also fit to bulk properties and to the Rose-Vinet equation of state. This was particularly important to ensure well behaved potentials where the lattice parameters were too small or large for DFT calculations to run and converge successfully. The energy predicted by the Rose-Vinet equation of state for these lattice parameters added to the dataset.

Several computer codes were developed. These allowed the automated convergence of parameters for the DFT calculations and computation of the Birch-Murnaghan equation of state and elastic constants using QuantumEspresso. A bespoke code was developed to fit EAM and 2BEAM potentials to this data.

At this stage, the potentials are only binary alloys. It would be desirable to extend these to include Ni, as its inclusion in the alloy causes the austenitic structure of the steel. Following this, Cr should be included as it is another important alloying element responsible for the corrosion resistance of the steel. This would perhaps necessitate moving from an EAM potential to a 2BEAM potential. This would have been too complicated and time consuming to complete in this work, and the DFT calculations would have been too complex for the resources available. Hence the restriction to binary alloys.

The above highlights the need for Fe-Pd and Fe-Ru fit to FCC DFT data for modelling radiation damage of austenitic steels doped with PGMs, and this work fills that gap. It also paves the way to more complex potentials required to better model radiation damage of austenitic stainless steels.

7.1.2 Summary of Chapter

The potentials require DFT generated force and energy data. Similar to calibrating experimental equipment, the parameters used in the calculations are converged to ensure the desired accuracy whilst maintaining the feasibility of the calculations (7.2). A database including perfect, perturbed, distorted, defect, bulk and slab configurations is created (7.3).

The development of a potential fitting code is discussed including methods used to help reduce the computational time (7.4). The choice of potential forms, DFT data used and how well the potentials fit the data are covered (7.5). The Rose-Vinet plots are compared, along with the DFT computed bulk properties and surface energies, to those created by these potentials. Finally a contribution to the DL_POLY MD is detailed (7.6).

7.1.3 Interatomic Potential Fitting

The process follows aspects of previous work. The force matching method of Ercolessi and Adams[145] and its application within Brommer's PotFit code[152] (and it's subsequent extension by Sheng[153]) impacted this work greatly. The exact fitting methods used by Bonny and spline type functions used throughout work on Iron alloys by Ackland, Mendelev, Hepburn and more helped to fix on a specific form of the functions and functional used here. The fitting process followed here is shown in fig. 7.1.

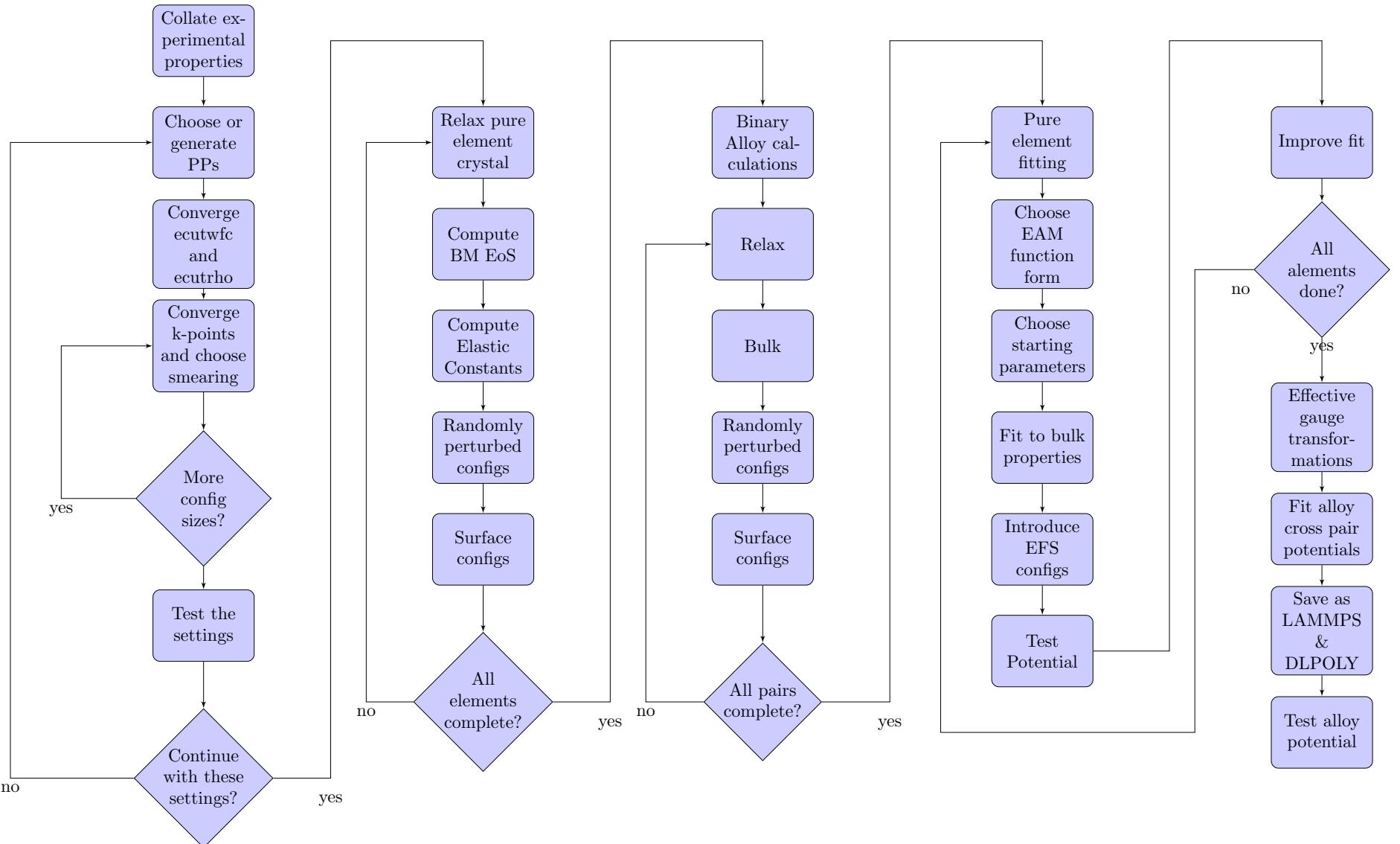
7.1.4 Existing Experimental & DFT Bulk Property Data

The bulk properties for Fe (BCC) and Pd are available on several websites and in published work. The lattice parameter, bulk modulus and elastic constants are available for Fe BCC and Pd FCC, but as pure Fe FCC is theoretical under normal conditions, this is computed here using DFT.

Element	Atomic Mass	Density kg/m ³	Atoms/m ³	FCC (Bohr/Angstrom)	BCC (Bohr/Angstrom)
Al [207]	26.98	2700	6.02×10^{28}	7.66/4.05	6.08/3.22
Cr [208]	52.00	7140	8.27×10^{28}	6.89/3.64	5.47/2.89
Fe [208]	55.84	7874	8.47×10^{28}	6.83/3.61	5.42/2.87
Ni [209]	58.69	8908	9.09×10^{28}	6.67/3.53	5.30/2.80
Ru [210]	101.07	12270	7.42×10^{28}	7.14/3.78	5.67/3.00
Pd [211]	106.42	12023	6.83×10^{28}	7.34/3.88	5.83/3.08

Table 7.1: Predicted lattice parameters based on the density, atomic number and type of structure

Figure 7.1: Work flow used to fit potentials



A large amount of computing power is required to run the DFT calculations. If the input parameters are not accurate enough to begin with, it will either take a long time for the DFT calculation to run, or it will fail to converge completely. The lattice parameter for each element, using the density of that pure element in its state under normal conditions, is predicted for that element either for both the FCC and BCC structures (table 7.1).

Aluminium appears here because it has a simpler electronic structure and no magnetic properties to complicate the DFT calculations. These calculations for Aluminium complete much faster than for the other elements listed, so it was used throughout to develop and test the computer codes created.

7.2 Selected DFT Code, Pseudopotentials and Parameters

7.2.1 Quantum Espresso

There are a choice of several different DFT programs to choose from, including VASP, Siesta, Castep, Abinit, and Quantum Espresso. The open source Quantum Espresso includes the PWscf binary that solves electronic structure calculations with plane wave pseudopotentials. It calculates the total energy of a system as well as the forces between atoms and the stress on the simulation box.

As the calculations contain Iron, it is important to take into account the magnetism caused by the unpaired filling of the d-shell. PWscf gives three modes: non-magnetic, collinear spin and non-collinear spin.

7.2.2 Pseudopotential Selection

The pseudopotentials were downloaded from the Quantum Espresso PSLibrary with URL:

<http://www.quantum-espresso.org/pseudopotentials/pslibrary>

There are several categories of pseudopotential available. The pseudopotentials may be fully relativistic or not. The DFT calculations in this work will be non-spin or collinear spin and the element with the largest number of electrons will be Palladium, with electrons in the s, p, d shells. There will be no elements used in this work with f shell electrons, so a non-relativistic pseudopotential will be used for each element.

There are a number of choices of exchange-correlation functional, depending on the element:

- pz: Perdew-Zunger (LDA)
- vwn: Vosko-Wilk-Nusair (LDA)
- pbe: Perdew-Burke-Ernzerhof (GGA)
- blyp: Becke-Lee-Yang-Parr (BLYP)
- pw91: Perdew-Wang 91 gradient-corrected

In the literature, LDA and BLYP are used for organic DFT calculations. LDA and GGA type pseudopotentials have been used to model solids and, in particular, the GGA type have been used to study metals as they are more reliable at calculating parameters such as a_0 (see section 5.7.11). Several elements were tested to compare the results of LDA and GGA pseudopotentials to the experimental values for the lattice parameter of each.

The settings used were:

- ecutwfc: 71 Ry
- ecutrho: 430 Ry

- k-points: 9 9 9 Monhurst-Pack grid offset
- smearing: 0.04 Ry Mazari-Vanderbilt cold smearing
- calculation type: non-polarised calculation
- pseudopotentials: PZ for LDA, PBE for GGA

Element	Crystal	a_0 exp. (ang)	a_0 LDA (ang)	a_0 GGA (ang)
Al	FCC	4.05	3.98	4.04
Fe	BCC	2.87	2.69	2.75
Pd	FCC	3.89	3.84	3.93
Cu	FCC	3.61	3.49	3.59

Table 7.2: Predicted lattice parameters based on the density, atomic number and type of structure

The computed results are given in table 7.2. The simpler LDA pseudopotentials consistently underestimate the lattice parameter a_0 for the four metals tested, and on average they underestimate by 3%. The GGA calculations are within 1.5% of the experimental value.

7.2.3 Smearing Type

The material being studied here is a metal/metal alloy, and this has important consequences when setting up the calculation. The conduction and valence bands overlap in a metal, and the Fermi energy passes through this overlap. When the occupied states are summed, integrating over all k-points k[212], those bands that cross the Fermi energy will drop to zero. There are two solutions used here to reduce this abrupt drop off: increase the number of k-points, or add a smearing to smooth the occupation function. Previous work has given the following suggested smearing types and values as a start point.

- Degauss values 0.01Ry Methessel-Paxton [213]
- Fermi-Dirac 0.01eV (0.00074Ry) [214]
- Marzari-Vanderbilt 0.01Ry [215]
- Degauss 0.03 + 0.05 (no units or scheme given, but other units in article are Ry) [216]
- Marzari-Vanderbilt 0.05Ry [217]

This suggests that a range between 0.01Ry to 0.05Ry should be investigated as a sensible range of smearing energies. There are a choice of four smearing functions in QuantumEspresso.

- Gaussian
- Fermi-Dirac
- Methfessel-Paxton
- Marzari-Vanderbilt

Functions such as the Gaussian and Fermi-Dirac are not cold smearing and will converge to the wrong energy whereas the Mazari-Vanderbilt and Methfessel-Paxton are designed to reduce artificially increasing the final energy whilst smearing the electron wavefunction. An author of QuantumEspresso[177], N. Marzari, co-developed the cold smearing Marzari-Vanderbilt function[180], and this was selected as it had appeared a number of times in the literature and throughout the QuantumEspresso tutorials.

7.2.4 Parameter Convergence

The energy cutoff, charge density, k-point and smearing values must be selected carefully for any similar DFT calculation. There is a balancing act between the accuracy of the result of a particular calculation and computational time. This will also vary with the pseudopotential, complexity of the electron configuration of the atoms, whether or not the calculation is with or without spin, and so on.

The pseudopotentials and smearing type have already been selected, and the first set of calculations will not include magnetism. The parameters must work well with high symmetry geometries, such as a BCC or FCC crystal with the atoms in their exact position, but also where the atoms have been slightly perturbed from their exact position in the lattice.

Ecutwfc and Ecutrho

A primitive FCC or BCC is created and the atoms are slightly perturbed, breaking symmetry but also putting forces on each atom within the cell. A reasonably high number of k-points are used and the smearing value should be low, but the k-point values in particular will depend upon the computing resources available.

Select a low ecutwfc and set the ecutrho equal to four times the value of the ecutwfc. Depending on the pseudopotential, the calculation may fail because the values are too small. If this is the case, continue increasing the value in 5Ry increments until the calculation runs successfully.

Continue to increase the ecutwfc value whilst increasing the value of ecutrho such that $\text{ecutrho} = 4 \times \text{ecutwfc}$ in steady increments (5Ry). Once the calculated force and energy per 1Ry increment fall below a threshold set by the user, the values have converged. In this work, the convergence values were:

- Energy convergence per 1Ry: 1.0×10^{-6} Ry (1.36×10^{-5} eV)
- Force convergence per 1Ry: 1.0×10^{-5} Ry/Bohr (2.57×10^{-4} eV/Ang)

At this point, it may be possible to reduce the ecutrho value further while remaining within this convergence threshold, so the ecutwfc value is held fixed while the ecutrho value is reduced in steps of 5Ry.

An attempt is made to reduce the value of ecutwfc further in 1Ry steps while holding the ecutrho value steady for as long as the convergence value remains within the set threshold.

K-Points and Smearing Value

A configuration with a similar size and number of atoms as those in the calculations should be created. In this work, the reference database and bulk properties will be calculated using 32 atom configurations in an approximately sized $7 \times 7 \times 7$ angstrom box. The Iron FCC k-points and smearing are calculated using a 32 atom FCC box with sides length 7.2 angstrom and the Palladium with a 32 atom FCC box with sides length 7.8 angstrom.

As with the ecutwfc and ecutrho convergence calculations, the atoms are slightly perturbed from their exact crystal positions, putting a force on the atoms and breaking the symmetry (as will be the case for the configurations used for fitting).

The convergence process is more complex for k-points and smearing. There might be a convergence that, at face value, looks better (a smaller convergence value and a decrease in the computation time and memory requirements) for combinations with a higher smearing value, but it is preferred to have a smaller smearing value with a higher number of k-points, while maintaining a reasonable computing time, as the greater the

smearing the further the calculated value is from the actual calculated value (with no smearing and a very high number of k-points).

A 2D plot of energy and force values as a function of number of k-points and smearing may then be used to choose reasonable settings that balance the requirements.

7.2.5 QECONVERGE Python Code

The convergence process requires the creation of many input files, and manually editing these files is time consuming and vulnerable to human error. Over 100 input files may be required to converge the ecutwfc and ecutrho values, with a further 50-100 required to generate the 2D k-point and smearing plots. A python code, QECONVERGE, was developed to automate this process and plot the results automatically for the user (section K).

The program reads an input file into memory, and this contains the settings for the convergence run. A template PWscf input file is also loaded, and this will detail any required PWscf settings, such as the pseudopotentials, atom species and so on.

Once these files have been loaded, the first stage of the program runs to determine the ecutwfc and ecutrho values that are within the force and energy convergence thresholds provided by the user. The program creates a crystal based on the settings, for example an FCC, BCC or SC crystal supercell. The exact atomic positions are likely to be the optimal, relaxed, positions and so the overall forces on each atom will be zero.

The program randomly varies the atomic positions, and this results in a configuration with forces between the atoms. The maximum amount each atom is perturbed from the exact lattice position is set in angstrom and is determined by the user. The perturbation is this maximum multiplied by a random float (on the range 0.0 to 1.0) picked from a Gaussian distribution. The random seed is set by the user to ensure repeatable calculations. In this work, the maximum perturbation was set to 0.01 angstrom in order to avoid issues with SCF convergence within PWscf.

The wfc value starts with the user defined ecutwfc starting value, and this is increased until the change in energy and force both converge to within the specified thresholds. The ecutrho value is increased simultaneously, and is four times the ecutwfc value. In the second stage, the ecutrho value is decreased and the ecutwfc value is fixed while the energy and force remain within their respective convergence thresholds. Finally, the program attempts to decrease the ecutwfc value further.

Following the energy and charge density cutoff, the program then runs a number of calculations in order to produce a collection of colour density plots of energy cutoff vs density cutoff, plotting the convergence of total energy and total force. This allows the user to visualise the convergence surface and select cutoff values manually, if desired.

The final part of the program is to help the user decide upon degauss and k-point values. The converged ecutwfc and ecutrho values from the first part of the code are used, and the k-point start, end and increment amounts are set by the user, as is a list of smearing degauss values. 2D colour plots of force and energy convergence as smearing changes and k-point value changes are prepared and saved so the user may study these to determine the best combination for their application.

As mentioned, the user sets the random seed. This means that, although they are pseudo-rng generated, the convergence jobs are repeatable given the same random seed. Every successful PWscf calculation is logged and saved. If the program runs multiple times, the cached output files are used rather than recalculating every input file, and this saves the code from repeating calculations that have already been completed.

The source code and instructions on how to use the program are available to download from GitHub.

<https://github.com/BenPalmer1983/qeconverge>

7.2.6 Convergence of Key DFT Settings and Testing

Prior to generating a database of atom configurations, calculated forces, stresses and energies, key DFT settings needed to be selected and tested. The Python code QECONVERGE was written then used to converge the ecutwfc, ecutrho, k-points and degauss smearing values for each of the pseudopotentials required.

As discussed previously (section 5.7.11) the GGA type pseudo-potential more reliably calculated the bulk properties of a material when compared to LDA/LSDA, and a copy of the earlier table is given in table 7.3.

Pseudo-potential	Element	a/angs	B_0/GPa
Experimental	Na	4.29[175]	6.3[175]
Na.pz-spn-kjpaw_psl.1.0.0	Na	4.06	8.72
Na.pbe-spn-kjpaw_psl.1.0.0	Na	4.20	7.67
Na.pbesol-spn-kjpaw_psl.1.0.0	Na	4.17	7.50
Experimental	Al	4.05[176]	76[176]
Al.pz-nl-kjpaw_psl.1.0.0	Al	3.98	78.6
Al.pz-n-kjpaw_psl.1.0.0	Al	3.98	78.6
Al.pbe-nl-kjpaw_psl.1.0.0	Al	4.04	91.3
Al.pbe-n-kjpaw_psl.1.0.0	Al	4.04	75.0
Al.pbesol-nl-kjpaw_psl.1.0.0	Al	4.01	87.7
Al.pbesol-n-kjpaw_psl.1.0.0	Al	4.01	79.0

Table 7.3: Experimental vs LSDA vs GGA - the DFT values were computed with a PWscf[177] using a 2x2x2 cell, 7x7x7 kpoints and ecutwfc=50

The following pseudopotential files were used for each element:

- Aluminium: a plane wave GGA potential Al.pbe-n-kjpaw_psl.1.0.0.UPF
- Chromium: a plane wave GGA potential Cr.pbe-spn-kjpaw_psl.1.0.0.UPF
- Iron: a plane wave GGA potential Fe.pbe-spn-kjpaw_psl.1.0.0.UPF
- Palladium: a plane wave GGA potential Pd.pbe-spn-kjpaw_psl.1.0.0.UPF
- Ruthenium: a plane wave GGA potential Ru.pbe-spn-kjpaw_psl.1.0.0.UPF

For each pseudopotential, the ecutwfc, ecutrho and kpoint and degauss settings were converged. Ideally, the k-points settings would have been increased, but doing so would have been prohibitive due to the amount of time taken to perform each calculation.

By the time calculations were being executed for a 32 atom supercell, one node of 20 processor cores would run for several days up to a week or more, requiring the calculation to be submitted to the job queue multiple times. Al is also considered in the results section, as it is a simpler atom to run trial calculations with, and to compare to experimental results.

The relaxed crystal lattice parameters were calculated for Fe, Cr, Pd and Ru. Cr will not be used in either the Fe-Pd or Fe-Ru fitting calculations. However, there is a large increase in computation time between non-spin and collinear spin calculations and the ferromagnetic structure of BCC Fe as well as the antiferromagnetic structure of FCC Fe and BCC Cr were investigated.

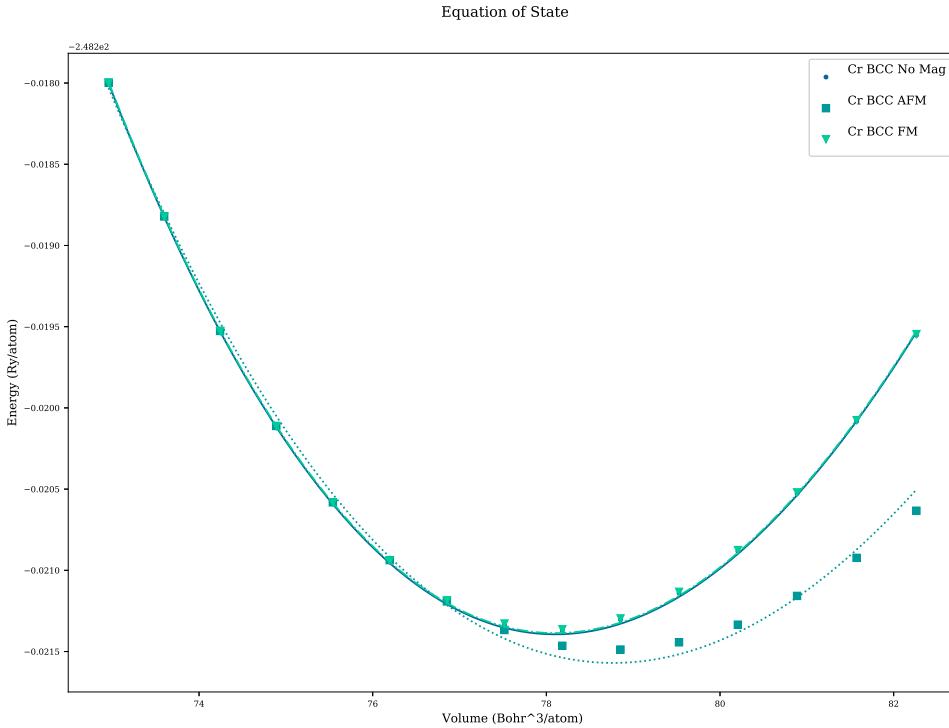


Figure 7.2: Equation of state fit through data points for Chromium BCC with no magnetism, ferromagnetic and anti-ferromagnetic configurations

Description	A_0 (ang)	V_0 (bohr ³)	B_0 (GPA)	E_0 (Ry) (DFT Only)
Exp.	2.91	83.23	160	-
AFM	2.86	78.77	217	-248.2216
FM	2.85	78.10	267	-248.2214
No Mag	2.85	78.102	267	-248.2214

Table 7.4: Chromium properties with and without collinear spin

The equation of state was calculated for Chromium BCC in three ways: (1) with magnetism switched off, (2) atoms set in a ferromagnetic configuration (collinear, all in the same direction), (3) atoms set in an antiferromagnetic configuration (collinear, atoms in the same cell with spin in opposing directions).

The DFT calculated values for the bulk modulus were larger than expected, but the antiferromagnetic calculation was much closer to the experimental value (table 7.4). The E_0 values do not reflect the actual energies, but show the relative calculated values. The antiferromagnetic configuration gives the lowest, optimum, energy (fig. 7.2).

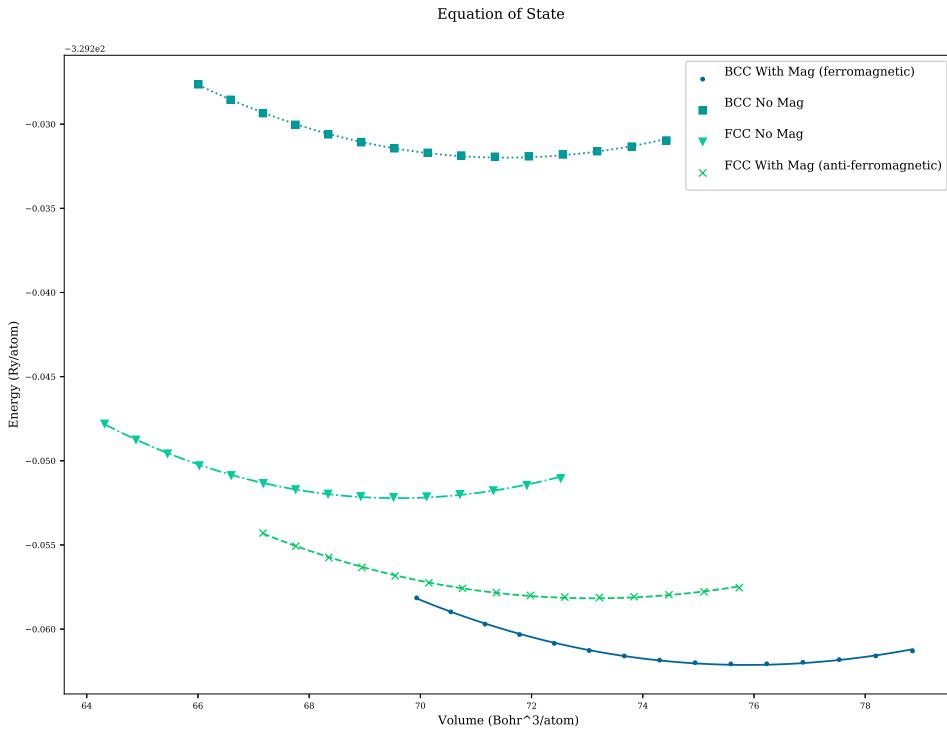


Figure 7.3: Equation of state fit through data points for Iron BCC (no magnetism, ferromagnetic) and Iron FCC (no magnetism and antiferromagnetic)

Description	A0 (ang)	V0 (bohr ³)	B0 (GPA)	E ₀ per atom (Ry) (DFT Only)
BCC Exp.	2.86	79.01	170	-
BCC No Mag	2.77	71.53	286	-329.23
BCC FM	2.82	75.9	239	-329.26
BCC AFM	(failed)	(failed)	(failed)	(failed)
FCC Exp.	(none)	(none)	(none)	(none)
FCC No Mag	3.43	69.59	286	-329.25
FCC FM	(failed)	(failed)	(failed)	(failed)
FCC AFM	3.42	73.11	226	-329.26

Table 7.5: BCC and FCC Iron properties with and without collinear spin

In the case of Iron, both BCC and FCC configurations were used. The antiferromagnetic calculation for BCC Fe and ferromagnetic calculation for FCC Fe failed in that either the SCF calculations failed to converge or there were other issues. As expected, the optimal configuration was BCC with ferromagnetism, and second to this was FCC with antiferromagnetism.

Despite the increase in computational time, and other demands on resources such as scratch space and RAM per node, it was clear that collinear spin calculations were required for iron. The lattice parameter value was within 2% of the experimental value with the BCC structure set in a ferromagnetic state. Whilst the bulk modulus is only within 41% of the experimental value, this is an improvement over the non magnetic calculation that was almost 70% away from the experimental bulk modulus (table 7.5, fig. 7.3).

The QECONVERGE program was used to suggest the parameters to use, given the convergence threshold of 1.0^{-5} Ry/Bohr for forces and 1.0^{-6} Ry for energy.

Element	Pseudopotential	Ecutwfc (Ry)	Ecutrho (Ry)	K-points	Degauss (Ry)	Atoms
Al	Al PBE KJPAW	50	200	11	0.04	32
Fe/Pd/Ru	PBE KJPAW	71	431	9	0.04	32
Fe/Pd/Ru	PBE KJPAW	71	431	Gamma	0.04	128-256

Table 7.6: DFT Settings - pseudopotentials, ecutwfc, ecutrho, k-points, smearing

These settings in table 7.6 were used throughout the remainder of the work. The maximum values were selected from Fe, Ru and Pd as they were to be used for calculations of the pure elements and for binary alloys of Fe-Ru and Fe-Pd.

The other settings used in PWscf throughout are listed in table 7.7. Several of these values were arrived at by trial and error. It was recommended by the authors of Quantum Espresso to adjust parameters such as the mixing beta if a calculation fails to converge. The mixing mode was also changed from time to time, but overall the plain mode seemed to work best in most cases.

Parameter	Value
etot_conv_thr	0.0001 Ry
forc_conv_thr	0.001 Ry/Bohr
conv_thr	1.0×10^{-8} to 1.0×10^{-6} Ry
diagonalization	david (Davidson iterative diagonalization)
mixing_beta	0.1 to 0.3
mixing_mode	plain (local-TF for defects or randomised configurations)

Table 7.7: DFT Settings - other parameters explicitly set in PWscf

The choices of parameters are supported by the convergence plots produced by the QECONVERGE code in figures L.1 and L.2 (appendix L).

7.3 DFT Reference Database and Bulk Properties

The potentials in this work were designed for austenitic stainless steel and the properties of the individual elements in the FCC structure were computed using DFT. This is due to the bulk properties only being available experimentally for FCC Pd. A database of atom configurations along with atom positions, energy, forces and stresses was also compiled using DFT calculations.

7.3.1 QEEOS Python Program

A Python code, QEEOS, has been developed to calculate bulk properties of a structure using the DFT code Quantum Espresso. It is able to fit the equation of state for a cubic structure, and elastic constants for orthorhombic structures (which includes the subset of cubic structures).

A code was developed to automate the process of calculating the equation of state and elastic constants of a material using Quantum Espresso. It requires an input file and a starting PWscf input file.

- the input configuration is relaxed using the vc-relax option in PWscf
- configuration files are created to compute the equation of state, and PWscf is used to calculate the energies of these configurations

- to compute the elastic constants, nine distortions are applied to the relaxed configuration with the required number of steps for each strain applied to each distortion, and these are processed with PWscf
- once all DFT work has completed, the Birch-Murnaghan equation of state is fit to the first set of energies, and the nine orthorhombic elastic constants are fit to the results of the second set of energies

The source code and instructions on how to use the program are available to download from GitHub.

<https://github.com/BenPalmer1983/qeeos>

The readme file along with a number of the functions used in the code are included in appendix M.

7.3.2 DFT Calculations of Bulk Properties

Although experimental data does exist for several of the structures, it was decided to calculate all the properties using DFT. This was partly to confirm that both the QEEOS and PWscf, with the converged settings, were giving reasonable values and also to give data points that would better align with the energy, force and stress configuration calculations.

Experimental data is not available for FCC Fe and FCC Ru and so it was essential to compute these. Although spin collinear equations were used throughout, it was particularly important for these settings to be used for Fe. The relaxation step allowed for a relaxation of the atom positions, the basis vector and lattice parameter. This gave cubic structures for Ru and Pd but a tetragonal structure for Fe.

The bulk property calculations were automated with the QEEOS code and this applied the preprogrammed nine strains, as discussed in section 5.2.6, and the results are discussed here.

7.3.3 Relaxed Crystal Calculations

The cohesive energy is an important value in relation to the interatomic potential. The energies calculated by DFT code depend on many factors including the energy under which plane-wave are cut-off, the degauss value, k-point settings and the SCF convergence parameters. What is more important in the DFT calculations is the difference in energy between calculations. As the experimental cohesive energy is known for certain elements, a DFT calculation may be performed for each species of atom to determine the relaxed energy. This value may then be used to calibrate other DFT calculations, so they are given relative to the energy of each atom spaced infinitely far from one another, with a binding energy of 0eV. The relaxed energies, the settings used to calculate them and the adjustment to convert them to the “real” energies are given in table 7.8.

Measurement	Al FCC	Fe BCC	Fe FCC	Pd FCC	Ru HCP	
Element	Aluminium	Iron	Iron	Palladium	Ruthenium	Ruthenium
Structure	FCC	BCC	FCC	FCC	HCP	FCC
Ecutwfc (Ry)	50	71	71	71	71	71
Ecutrho	200	430	430	430	430	430
K-points	11 11 11	9 9 9	9 9 9	9 9 9	9 9 9	9 9 9
Smearing (Ry)	0.04	0.04	0.04	0.04	0.04	0.04
NSpin	0	2	2	0	0	0
Starting Magnetism	None	FM	AFM	None	None	None
No. Atoms	32	16	32	32	16	32
Energy (Ry)	-1264.08749654	-5268.18846365	-10536.25040753	-16384.95803030	-6906.78722345	-13813.30088106
Energy/Atom (Ry)	-39.502734267	-329.261778978	-329.257825235	-512.029938447	-431.674201466	-431.665652533
Energy/Atom (eV)	-537.462275214	-4479.836349416	-4479.782555982	-6966.524743211		
Known Cohesive Energy (eV)	-3.36	-4.316	-4.26 (Calculated)	-3.91	-6.74	-6.62 (Calculated)
Adjustment/atom (eV)	534.102275214	4475.520349416	4475.520349416	6962.614743211	5866.47338	5866.47338
Relaxed a_0 (Bohr)	15.265	10.594	12.937	14.845		
Relaxed c (Bohr)	None	None	None	None	1.5773	None
U_{xx}	1.000	1.000	1.000	1.000	1.000	1.000
U_{yy}	1.000	1.000	1.054	1.000	0.8660	1.000
U_{zz}	1.000	1.000	1.000	1.000	1.5773	1.000
U_{yx}	0.000	0.000	0.000	0.000	-0.5	0.000

Table 7.8: Relaxed energies calculated in Quantum Espresso - U_{nn} is the entry in the 9 element matrix that represents the three basis vectors of the cell, where values not specified here are equal to zero.

The configuration for Fe has been referred to throughout this work as FCC, but the relaxed collinear spin-polarized DFT calculation predicts that the structure is slightly face centered tetragonal (FCT). The optimal magnetic setting is antiferromagnetic and, with the spin aligned in the z-direction, the y-axis of the unit cell is approximately 5% larger than the x and z axis.

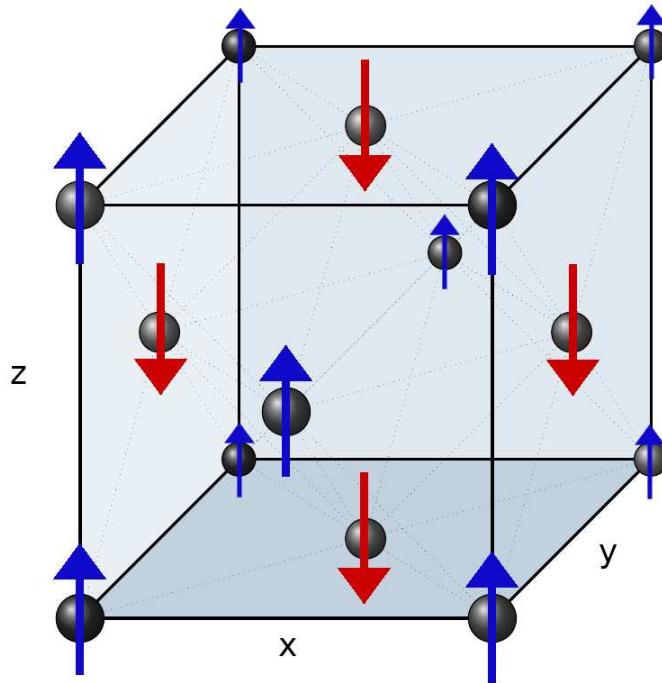


Figure 7.4: Magnetic alignment along the z-axis, antiferromagnetic Fe FCC

The atoms in the (0,1,0) plane are majority spin-up electrons, along the z-axis, and those in the (0,2,0) plane are spin-down, along the z-axis (fig. 7.4). Following the relaxation calculation, all the atoms were in their standard FCC positions, and the magnetic moment for each atom was 1.4901 Bohr magneton and -1.4901 Bohr magneton for the spin up and down respectively.

7.3.4 Calculation of Elastic Constants

The planewave cutoff and k-points were converged to within the required parameters for energy and force but it is known that, in general, LDA and GGA pseudopotentials under and over estimate the lattice parameters. The bulk modulus and elastic constants were calculated for FCC Aluminium and BCC Iron in order to compare to experimental values.

A summary of the main data points is given here for FCC Al, BCC Fe, FCC Pd and FCC Fe, and the first three listed are compared to experimental data. A full details of the calculated properties including plots are included in appendix N.

FCC Aluminium

The parameters used for the Al calculations were arrived at earlier using the convergence code. The remainder of the parameters are either default settings, or were selected following trial and error (for example, reducing the mixing beta to help achieve convergence). These settings are given in table N.1.

The results in table N.2 show a good agreement between the experimental values for Al and the calculated values. The lattice parameter is a particularly good fit, being within 1% of the experimental value. The shear modulus does not match as well, but is within 20% of the experimental value. The RSS measurement of fit between the values was 2.54×10^2 .

BCC Iron

The equation of state and elastic constants were calculated for BCC Fe, with no-spin and collinear spin. The settings used for the calculations are given in table N.3.

The non-magnetic Iron calculation was particularly poor. Although the lattice parameter was within 5% of the experimental value, the structure was unstable and had negative values for both the Young's modulus and shear modulus (table N.4). The RSS measurement of fit between the values for the was 2.01×10^5 .

With collinear spin enabled, the structure is stable. The lattice parameter is still slightly under the experimental, but is now within almost 2% of the value. The bulk modulus predicted by fitting the Birch-Murnaghan equation of state is more than 40% the experimental value, but the Reuss and Voigt values derived from the calculated elastic constants are just 20% difference to the experimental value. The latter two values are calculated using the elastic constants, rather than the equation of state. This highlights the importance of using collinear spin calculations, despite the cost in computing time. The RSS measurement of fit between the values was 1.09×10^4 giving an improvement of at least one magnitude.

FCC Iron

The gamma phase of pure iron does not exist for experimental measurements to be made, and the results here will be used to fit the FCC iron potential in section 7.5. The settings used for the calculations are given in table N.5.

The structure is stable under the orthorhombic stability conditions given in section I.2.5. Whilst there are no experimental measurements to check against, the values are sane when compared to the other calculations performed here (table N.6). The C_{44} , C_{55} and C_{66} values are slightly large when compared with those for BCC, but further DFT calculations would need to be performed to investigate this.

The results computed here are summarised in table 7.9 and these are used by the EAMPA code in section 7.5 to fit the potential for pure Fe.

Property	Value used in potential fitting
Element	Fe
Structure	Face Centered Cubic
a_0	3.59 Angstrom
Nearest Neighbour	1.85 Angstrom
Basis vectors	$\begin{bmatrix} 0.96 & 0.0 & 0.0 \\ 0.0 & 1.00 & 0.0 \\ 0.0 & 0.0 & 0.96 \end{bmatrix}$
E_{coh}	-4.32 eV
B_0 (GPA)	226.1
E (GPA)	356.8
G (GPA)	144.8
Poisson Ratio η	0.23
Elastic Constants	$\begin{bmatrix} 364.6 & 141.6 & 233.8 & 0 & 0 & 0 \\ 141.6 & 298.7 & 130.4 & 0 & 0 & 0 \\ 233.8 & 130.4 & 364.6 & 0 & 0 & 0 \\ 0 & 0 & 0 & 186.3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 266.8 & 0 \\ 0 & 0 & 0 & 0 & 0 & 186.3 \end{bmatrix}$

Table 7.9: DFT computed FCC Fe parameters for fitting the potentials in this work

FCC Palladium

The equation of state and elastic constants were calculated for FCC Pd with no-spin. The settings used for the calculations are given in table N.7.

There is a good agreement between the computed and experimental values for Pd, without the need to move from non-spin to collinear spin calculations (table N.8). However, in the alloy calculations the presence of Fe will dictate the necessity for collinear spin. The RSS measurement of fit between the values was 4.12×10^3 .

The results computed here are summarised in table 7.10 and these are used by the EAMPA code in section 7.5 to fit the potential for pure Pd.

Property	Value used in potential fitting
Element	PD
Structure	Face Centered Cubic
a_0	3.925 Angstrom [211]
Nearest Neighbour	Angstrom [211]
Basis vectors	$\begin{bmatrix} 1.0 & 0.0 & 0.0 \\ 0.0 & 1.0 & 0.0 \\ 0.0 & 0.0 & 1.0 \end{bmatrix}$
E_{coh}	3.91 eV [140]
B_0	184.4 GPA [140]
Elastic Constants	$\begin{bmatrix} 218.5 & 151.4 & 151.4 & 0 & 0 & 0 \\ 151.4 & 218.5 & 151.4 & 0 & 0 & 0 \\ 151.4 & 151.4 & 218.5 & 0 & 0 & 0 \\ 0 & 0 & 0 & 80.3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 80.3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 80.3 \end{bmatrix}$

Table 7.10: DFT computed FCC Pd parameters for fitting the potentials in this work

FCC Ruthenium

Ruthenium at standard conditions exists as hexagonal close packed, but as this work is focused on FCC steel doped with PGMs, the properties are computed for FCC Ruthenium. The settings used for the calculations are given in table N.9.

The results computed here are summarised in table 7.11 and these are used by the EAMPA code in section 7.5 to fit the potential for pure Ru.

Property	Value used in potential fitting
Element	RU
Structure	Face Centered Cubic
a_0	3.809 Angstrom [211]
Nearest Neighbour	Angstrom [211]
Basis vectors	$\begin{bmatrix} 1.0 & 0.0 & 0.0 \\ 0.0 & 1.0 & 0.0 \\ 0.0 & 0.0 & 1.0 \end{bmatrix}$
E_{coh}	6.624 eV [140]
B_0	307.8 GPA [140]
Elastic Constants	$\begin{bmatrix} 471.5 & 219.1 & 219.1 & 0 & 0 & 0 \\ 219.1 & 471.5 & 219.1 & 0 & 0 & 0 \\ 219.1 & 219.1 & 471.5 & 0 & 0 & 0 \\ 0 & 0 & 0 & 245.0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 245.0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 245.0 \end{bmatrix}$

Table 7.11: DFT computed FCC Ru parameters for fitting the potentials in this work

7.3.5 Calibrating the Cohesive Energies

Density Functional Theory solves the Kohn-Sham equations for an infinite repeating lattice, and at first it doesn't seem applicable to single atoms or molecules that are not part of much larger lattice. There are settings that may be used to compute the energy of an isolated atom and these are discussed on the Quantum Espresso forum[218] and Materials Square[219].

The settings of note used in these calculations were a configuration of just 1 isolated atom in a large cubic cell (20+ Bohr). The computations were all $n\text{spin} = 2$ to ensure the electrons are computed spin up and down. The `assume_isolated` setting is given the makov-payne setting, as the system is a cube.

It was important to reduce the number of processor cores as too many would cause matrix decomposition errors for a low number of atoms, causing Quantum Espresso to crash. An option to leverage more cores was to split the calculation into pools using the `-npool` input flag for PWscf, but this was not necessary for this set of calculations.

The isolated and relaxed energy values were computed and from these values the cohesive energies were calculated (fig. 7.5 and table 7.12). In doing so, it became apparent that there were errors of up to 12% between the known values and those computed with DFT.

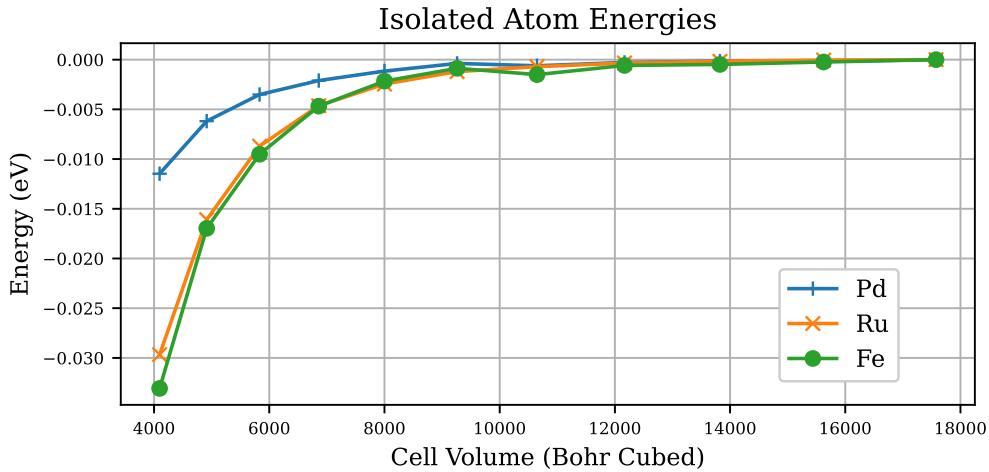


Figure 7.5: Isolated atom energies as cell size increase

An offset was needed to convert from the DFT energies to “real” energies that could be used to derive a potential.

Element	Structure	Isolated Energy (Ry)	Relaxed Energy (Ry)	Cohesive Energy (eV)
Fe	BCC	-4474.99115	-4479.83524	4.84 (exp. 4.32)
Fe	FCC	-4474.99115	-4479.78095	4.79
Pd	FCC	-6962.47261	-6966.52234	4.05 (exp. 3.91)
Ru	HCP	-5866.39894	-5873.22666	6.83 (exp. 6.74)
Ru	FCC	-5866.39894	-5873.11029	6.71

Table 7.12: DFT calculated energies per atom

Two offsets were computed and these depend upon the settings used in the calculation (table 7.13). Specifically, they depend on whether 9x9x9 k-points are used, in the case of the 32 atom calculations, or the gamma point is used (128, 129 and 256 atoms).

There were significant differences between the calculated cohesive energy, taking the difference between the isolated and bulk energies, and the experimentally known energies. Rather than use these all the DFT energies

Element	Reference Structure	Energy (Ry) (9x9x9)	Energy (Ry) (Gamma)	Cohesive (eV)
Fe	BCC	-329.2618197	-84290.00709480	-4.316
Pd	FCC	-512.0299524	-131079.09207327	-3.91
Ru	HCP	-431.6742014	-110507.45936537	-6.74

Table 7.13: Relaxed energies for 32 atoms, using 9x9x9 k-points, and 256 atoms, using the gamma point. These were used with the known cohesive energy to compute the offset energy.

in the remainder of this work were recalculated using the relaxed DFT energy and the known cohesive energy. This was to ensure the potentials were trained towards the known cohesive energies.

7.3.6 Compiling a DFT Reference Database

A database of configurations generated by DFT was compiled for Iron, Palladium, Ruthenium with alloys of Iron-Palladium and Iron-Ruthenium. The potentials are derived for FCC Iron and so more weighting was given to configurations that were also of this structure.

The QEEOS code was used to compute the bulk properties and in doing so it created PWscf input files and processed the output files. These output files are saved but the program and are as valid to use in the reference database as any. In total the database has over 600 configurations for Fe-Pd and almost 600 for Fe-Ru.

The distribution of atom separations was measured across the database and these are plotted in fig. 7.6. The full set is available to download from https://github.com/BenPalmer1983/fepd_feru_potential.

All the configurations in this work are FCC structure crystals. The QEEOS program, section 7.3.1 and appendix M, was used to compute the relaxed lattice parameter and primitive cell basis vector.

This program also automated the computation of equation of state, bulk modulus and elastic constants. A 32 atom supercell was used for each pure element where the bulk properties were computed. Ideally a large supercell or a larger number of k-points would have been used but throughout this work the number of processor cores per node ranged from 20-72 and this along with time constraints limits complexity of the calculations.

The data output from the QEEOS code forms a part of the reference database. A number of configurations used to make up this database are given in appendix U.

7.3.7 Pure Element Defects

The potentials are created to apply to FCC Iron, Palladium, Ruthenium and alloys of Fe-Pd and Fe-Ru. It was planned for the reference database to contain configurations with several defects:

- octahedral interstitial
- tetrahedral interstitial
- 100, 110 and 111 interstitials
- crowdion
- a vacancy

The configurations for these defects were estimated using the perfect crystal positions of 2x2x2 supercells containing 32 atoms. The defect or vacancy was applied, in some cases changing the total number of atoms, and the defect position was estimated.

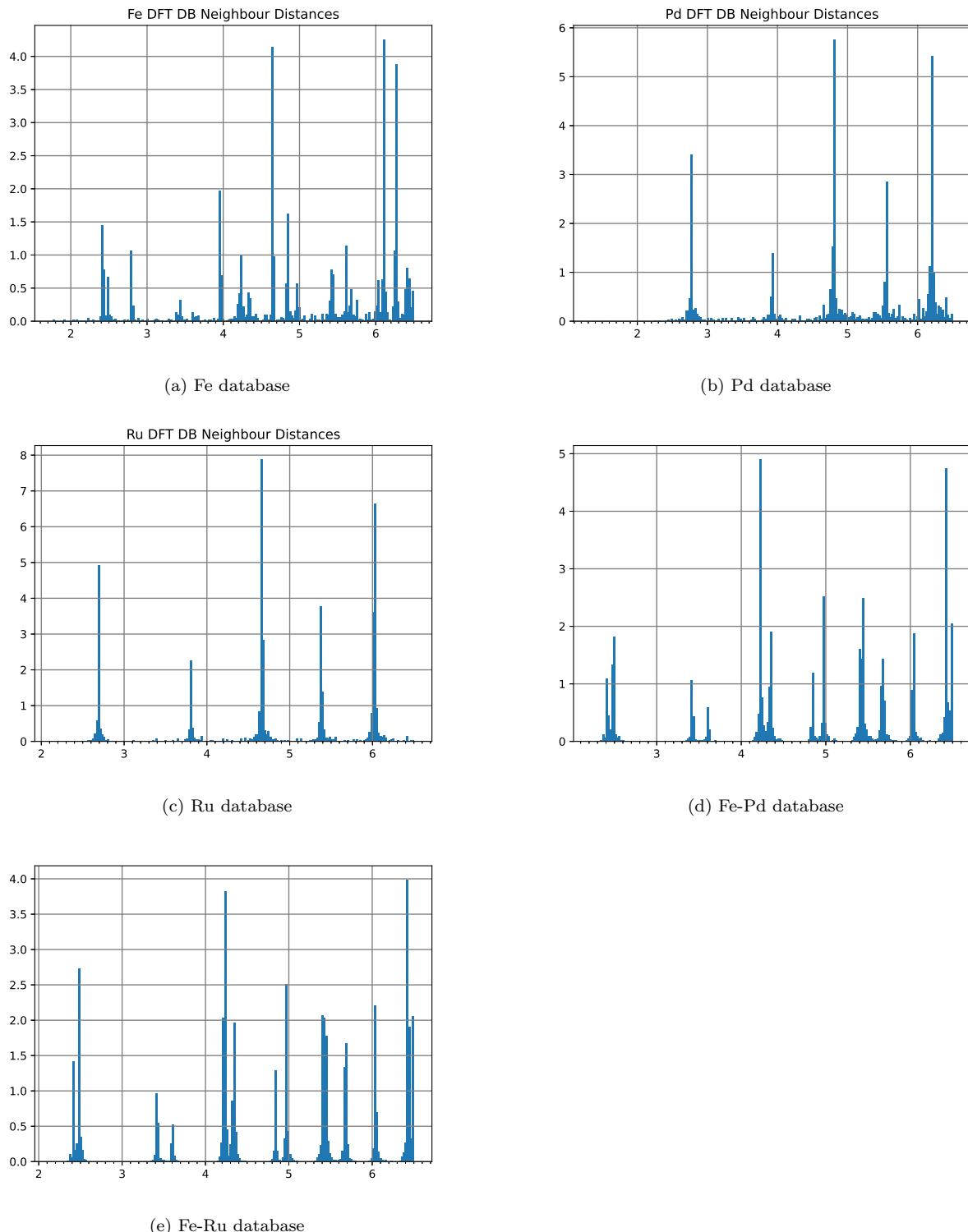


Figure 7.6: Neighbouring atom separations and their frequency over 5 subsections of the DFT reference database. The neighbour separation, in angstrom, runs along the x-axis. The neighbour separations were sorted into 200 bins for each reference database, and the percentage each bin is filled runs along the y-axis.

A number of calculations, and the chance of a successful convergence in both the electronic state SCF calculation and relaxation calculation, were very sensitive on the input parameters. Al was not as sensitive and was used to relax the starting configuration. The atom positions from the Al calculations were then used as the starting point for either Fe, Pd or Ru.

Due to the periodic nature of these calculations, having only 32 atoms was not ideal as each defect cell would be very close to another. Unfortunately, as stated a number of times, it was a constraint that was needed to be able to complete the work in a reasonable amount of time.

7.3.8 Alloy Atom Positions for Pd and Ru

The stainless steel being studied is to be doped with 1% or less of a platinum group metal. Ideally a super cell with at least 256 atoms ($4 \times 4 \times 4$ primitive FCC cells) would be used for all calculations, but due to the complexity of the calculations the majority were $2 \times 2 \times 2$ super cells containing just 32 atoms.

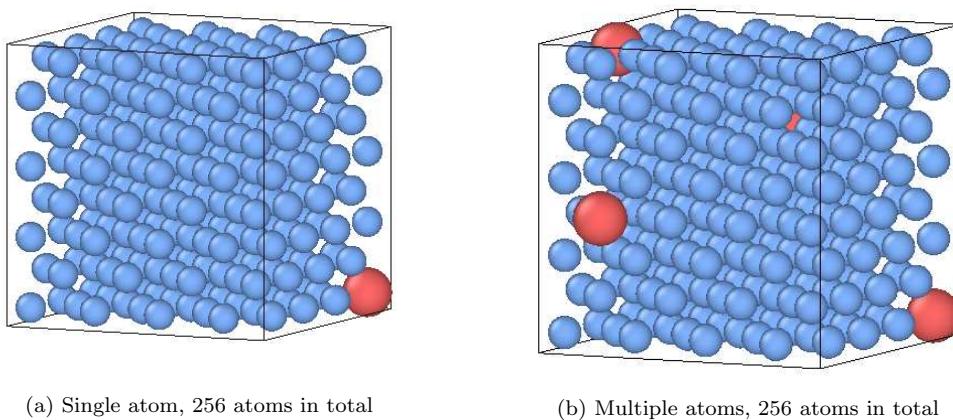


Figure 7.7: Binary alloy configurations with less than 2% PGMs

Most of the configurations were prepared using the relaxed FCT structure computed for Fe. One or more iron atoms were then replaced with either Pd or Ru. The majority of calculations were carried out using the 32 atom configurations. The larger 256 atom calculations were better suited to represent the low percentage of PGM but were only computed using the gamma k-point (fig. 7.7). This was another compromise in order to complete the calculations in time.

7.3.9 Configurations with Randomised Positions

Many of the atom configurations generated up to this point are symmetric and, whilst they have a variety of energies and stresses to use for fitting, the forces on the atoms are almost balanced and are small or none. A large set of configurations with positions perturbed slightly from the exact lattice position are created.

Rather than use a linear random distribution a Gaussian distribution is used. The majority of the atoms are closer to the exact crystal position but a minority are further and further away, as would be expected with this type of distribution. The maximum amount of perturbation was selected randomly between a minimum and maximum value. The effect on the calculation by making these perturbations was to increase the energy of the system as the maximum perturbation amount was increased, as can be seen in fig. 7.8.

As the amount at which the atoms are perturbed increases, the total force between the atoms also increases. If the disturbance increased too much, it will likely be useless for our needs as it would be beyond the temperatures relevant to this work, and it may also cause the SCF calculations to fail. When these calculations were processed the maximum perturbation being too large was the primary factor that caused PWscf to fail.

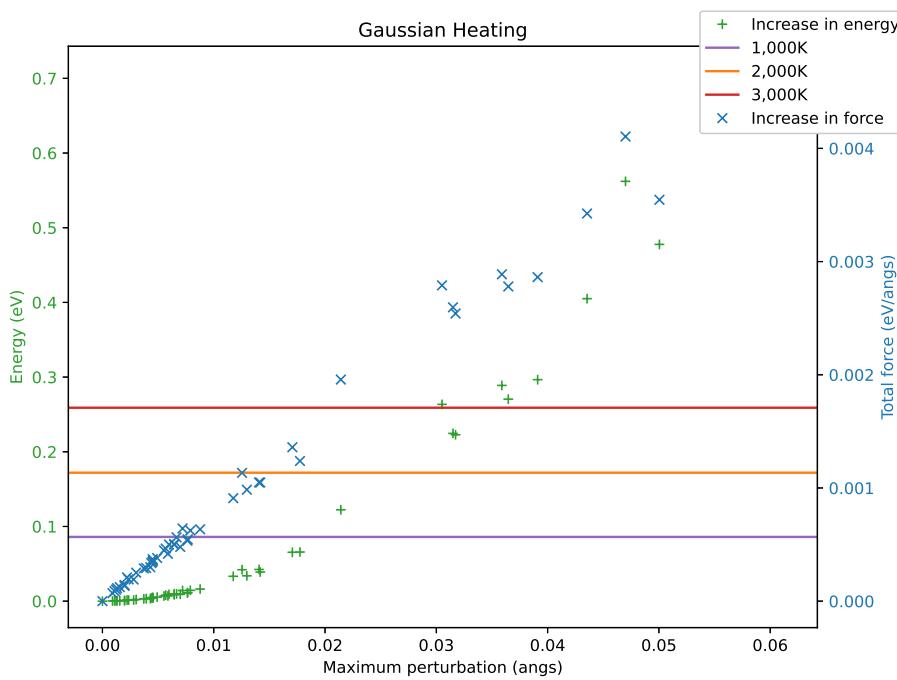


Figure 7.8: Effect of randomly perturbing atom locations

In the case of the alloy configurations, the randomised positions were loaded into the files. In all cases convergence failed. As perturbed atom positions had already been computed for Fe a new set of input files were created holding the Fe atoms in their perfect locations. The added Pd and Ru atoms were then perturbed slightly from their perfect position within the lattice in order to compute forces and energies at slightly different positions. Perhaps with more time and refined settings alloys with non-perfect atom positions could be calculated, but this would need to be investigated.

7.3.10 Surface Energy Calculations

One key aspect of this work is to determine whether or not either Pd or Ru are depleted from the surface at grain boundaries under irradiation. With this in mind, the surface energies for the pure elements were calculated by creating a slab of atoms with an open surface on each side.

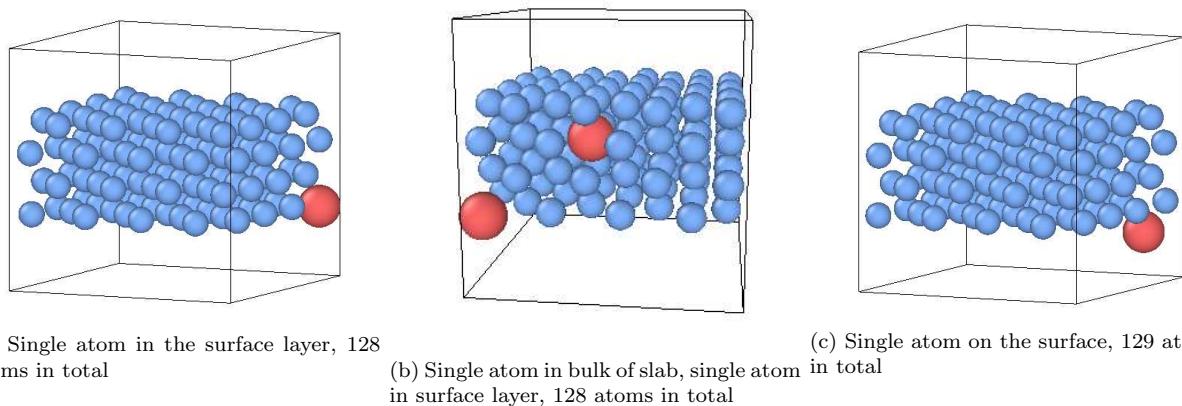


Figure 7.9: A sample of configurations used for binary alloy DFT calculations for Fe-Pd and Fe-Ru.

A set of 32 atom configurations were used where the separation between the two facing surfaces is varied. Several more 128 atom configurations were also prepared for the three pure elements. To investigate the effect

of a small addition of either Ru or Pd in the alloy a slab of Fe was prepared with Ru or Pd either within the slab, at the surface or on the surface (fig. 7.9). These examples were then added into the reference database used to fit the binary potentials.

7.3.11 Defects in Fe-Pd and Fe-Ru Alloys

Attempts were made to compute simple defects in the alloy but the majority failed to converge, either in the electronic structure SCF iteration or a failure to relax geometrically that then lead to a SCF failure. Rather than focus too much on this, more time was spent investigating Ru and Pd in a slab of Fe where the calculations were more successful.

7.3.12 DFT Errors, Convergence Failures and Repeat Calculations

A number of issues were encountered throughout this work when using DFT. Most failures resulted from either a lack of computational resources or failure to converge due to inappropriate starting configurations or PWscf settings. These are discussed in detail in appendix O where solutions to a number of the problems are discussed.

7.3.13 Summary of Configurations Generated with DFT

The total number of configurations that resulted from the DFT work was a smaller set than desired. Many of the defect calculations, particularly for Fe, failed to converge and were omitted from the database. A summary of those included is listed in table 7.14.

Configurations	Fe	Pd	Ru	FePd	FeRu
FCC bulk 256 atom (perfect)	1	1	1	1	1
FCC slab 128 atom	1	2	2	15	15
FCC random 32 atom	50	55	61	28	13
FCC slab 32 atom	10	25	17	0	0
FCC EoS 32 atom	7	16	15	0	0
FCC EC 32 atom	0	63	81	0	0
FCC defects 32 atom	0	64	14	0	0
FCC bulk 256 atom (random)	1	1	1	27	47
Total	70	227	192	44	76

Table 7.14: A summary of the configurations used for fitting the interatomic potentials. The Fe-Pd and Fe-Ru fitting also included the configurations for the pure elements, although they were not required when fitting the cross potential. Random configurations include those with slightly perturbed atom positions as well as those with slightly perturbed lattice parameters.

7.4 EAMPA: Potential Analysis and Fitting Code

7.4.1 Introduction

A python-fortran based computer code was developed to analyse interatomic potentials and fit their parameters to bulk properties and DFT calculated forces and energies. It was designed to take advantage of both Fortran and Python. Fortran is used to compute the neighbour lists, energies, stresses and forces, and Python is used to read and write data, produce plots and control the potential fitting. This allowed the use of custom optimization

subroutines as well as the use of those included in the SciPy module. Highlights from the code are discussed in this section and the appendix of this work (appendix S).

The latest version of the code has the following features:

- calculate energy, forces and stress of an ensemble of atoms
- calculate bulk properties of a potential for pure BCC or FCC structures
- relaxation of the lattice parameter
- relaxation of the cell geometry along with the lattice parameter
- fitting on analytic potential parameters using simulated annealing, genetic algorithm, Nelder-Mead, Levenberg-Marquardt, BFGS
- effective gauge transforms of pure element potentials
- creation of potential files in other formats (tabulated, LAMMPS etc)

The source code and instructions on how to use the program are available to download from GitHub.

<https://github.com/BenPalmer1983/eampa>

7.4.2 Potentials

The code has been designed specifically for many body potentials. The function types available for pair, density and embedding are listed in the appendix (T). More function types may be easily added by editing the module in the Fortran code that defines each function.

7.4.3 Calculations

The calculations involved in this code centre around the calculation of energy of configurations as well as the forces and stresses when necessary. All configurations are held within the Python program, but the neighbour list and energy (as well as stress and force) calculations are carried out in an imported Fortran module.

Energy, Force and Stress Calculations

A potential and configuration of atoms may be loaded into the program in a number of ways. Python passes this data to the atom Fortran module and then computes the neighbour list before it is passed back to the Python program where it is stored for later use.

There are a number of functions that allow the potential as well as atom configurations (and neighbour lists) to be updated. These are handled in Python, but the calculations are carried out in the Fortran atom module, that then passes the results back to Python. One example of such a function is where the position of the atoms is changed and the neighbour list is updated without a full recalculation.

The energy, forces and stresses may be computed for a range of configurations or a single configuration. This process starts in Python, but the calculations are once again completed using the Fortran module. The method used is discussed in section R.0.1 and the subroutine used is included in the appendix (section S.1).

Geometry and Coordinate Optimisation

Once a configuration is loaded into the program, it can be relaxed using a variety of techniques. The lattice parameter may be relaxed using the built in BFGS or CG minimizers and the same functions may be used to relax the simulation cell. Where the coordinates of the atoms are relaxed, a damped molecular dynamics subroutine is called from the Fortran module that computes the forces on the atoms and moves them accordingly. This has been more successful than using built in minimizers, where no gradient data is passed, but a more promising approach might be to include the VA08 subroutine from the Harwell archives[220]. This is the same subroutine used in the Bonny potential fitting code[139].

Bulk Property Calculations

The bulk property calculations rely on the calculation of the energy of a number of configurations where specific strains have been applied to the perfect relaxed crystal. The methods used are discussed in chapter 5 and the subroutine that creates the configurations, with which the module calculates the energies used to plot the equation of state and other distortions, is included in appendix S.

The Python code creates the configurations with the strains applied and the Fortran subroutines are called to compute the energies. Once the values are collected, the bulk properties are calculated in Python and any required plots are created.

As part of the calculation process the optimum (relaxed) lattice parameter is first calculated for the potential. Either a BFGS or CG minimizer is used. The basis vectors are fixed by the user, although there could be the option in the future to allow the basis vectors to also be adjusted. In this work the basis vectors of FCC Fe were fixed as the tetragonal determined by DFT in the input file for EAMPA.

During the fitting process, each time the potential is varied, the lattice parameter and bulk properties need to be recalculated. To save time the neighbour list is only calculated once, and the atom separations are recalculated based on the initial list, the recalculated lattice parameter and basis vectors.

RSS - How Well The Potential Fits Experimental & DFT Data

The fitting process measures how well a potential fits the data when the initial parameters are given and every time a new set of parameters are proposed. The user assigns weights to configurations and the parameters the potential is being fit to. The Python code calculates the residual squared sum (RSS) of the differences between calculated energies, forces, stress, lattice parameter, cohesive energy, bulk modulus and elastic constants and the known values. The sum of these values measures how well the potential fits.

Overview of the Potential Fitting Process

The input files are read in by Python that then iterates over the fitting process while changing the potential. The Fortran modules calculate the energy, force and stress of the loaded configurations as well as the bulk property calculations for pure crystals of the atomic species and structures specified by the user. To save on processing time, the neighbour lists for all configurations, including those used to calculate the equation of state and elastic constants, are only calculated once.

The step by step flow of the program while fitting parameters of a potential is as follows:

1. Read input files
2. Load Fortran Modules

3. Load potentials into Python
4. Load configurations into Python
5. Create neighbour list configurations in Fortran and save to Python
6. Begin fitting loop in Python
7. Loop starts - loop through potential function parameters
 - (a) Update potential in Python
 - (b) Create tabulated version of functions
 - (c) Run calculations in Fortran
 - (d) Loop starts - loop through configurations
 - i. Calculate energy, force, stress using pre-calculated neighbour lists
 - (e) End loop
 - (f) Save energies, forces and stresses back to Python
 - (g) Compute bulk properties from the energies in Python
 - (h) Calculate the RSS between the potential computed and known values
 - (i) Save parameters if best
 - (j) Make next set of parameters
8. End loop
9. Output results

Once the fitting process has completed, the best set of parameters are stored for the user along with a plot of the potential functions, equation of state and elastic constant energy-strain distortion plots (where these are used in the fitting process).

Global Optimization

The EAMPA code has two global optimization algorithms plus a number of other local and global optimization algorithms implemented through SciPy. These are discussed in greater detail in appendix Q.

The simulated annealing algorithm is implemented in the code and is an option available to the user for fitting. The algorithm looks for better solutions and replaces the current value with better values. It also accepts worse solutions with a probability that decreases with “temperature”.

A genetic algorithm has also been implemented. The user chooses the parameters for the algorithm and it attempts to search for the optimum parameters to fit the potential functions to the data.

The user settings include the following:

- population size
- “fresh” (new randomly generated) parameters
- number of generations
- how often the best results are enhanced with a chosen local optimization algorithm

Mixing Parameters from Parents to make Children

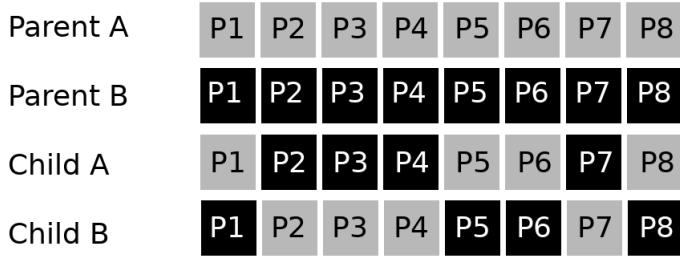


Figure 7.10: Parameter Breed Event

The function that breeds the parameters randomly combines the parameters from two parent solutions to create two children parameter sets. The parents are either both from the existing pool of solutions, or a parent from the existing pool with a parent from a fresh set, and the fresh set of parameters changes with each generation (fig. 7.10).

There is a chance that the child solutions will mutate slightly from the two parent combination, and there is also a guard that prevents cloning of solutions. A random selection of the best parameter sets are enhanced periodically using a local optimization method (CG, BFGS) and these are introduced to the pool of parameters.

Minimising Runtime

When fitting a potential, for each trial set of functions, the program must calculate the energy, forces and stress for each of the configurations and tens or hundreds of energy only calculations for the equation of state and elastic constant calculations. Python is a well used programming language in many areas of Science and it has the benefits of many modern languages. For certain specific tasks, it is slow in comparison to Fortran.

The energy, stress and force calculations were programmed in Fortran and use OpenMP to improve the calculation speed. This could be improved further by processing batches of configurations in parallel, but this isn't currently implemented. For now, a seed is picked so multiple instances can be run in parallel to give multiple optimised potentials to choose from.

The code optimises analytic potentials, but a number of the functions available are complicated and require the evaluation of many functions in order to give an output. To speed up the calculation, the user has the option to convert the analytic potentials to tabulated versions and these are interpolated using Lagrange polynomials (eq. 7.1 7.3).

$$D = \{(x_0, y_0) (x_1, y_1) (x_2, y_2) \cdots (x_n, y_n)\} \quad (7.1)$$

$$p_n(x) = \sum_{i=0}^n y_i L(i, x) \quad (7.2)$$

where $L(i, x) = \prod_{j=0, j \neq i}^n (x - x_j)$

There are typically hundreds or thousands of data points, so to interpolate, the closest few points are used. The

default number used in the code is 4 points. The gradient of the potential functions are also computed using Lagrange polynomials (eq. 7.3).

$$q_n(x) = \sum_{i=0}^{i=n} y_i g(i, x) \quad (7.3)$$

$$\text{where } g(i, x) = \left(\prod_{j=0, j \neq i}^{j=n} \frac{1}{x_i - x_j} \right) \times \left(\sum_{k=0, k \neq i}^{k=n} \prod_{j=0, j \neq k, j \neq i}^{j=n} (x - x_j) \right)$$

A pseudocode for both interpolation of the function points and derivatives are available in the appendix (section S.7 and S.8).

Effective Gauge Transforms

In order to create a potential for an alloy an effective gauge transform, as described in section 5.4.4, was applied to each pure potential. A mode added to the EAMPA program reads in the analytic potential and creates a numeric potential that has had the transform applied. This allows the pure potentials to be used together, although a cross pair potential is still needed.

LAMMPS Format

LAMMPS is a popular molecular dynamics code and the EAMPA code outputs potentials in its own format as well as the LAMMPS format. The structure of the file depends on the number of elements and there are a number of caveats to note including the pair potentials being stored in the file as the value multiplied by the distance, in angstrom[221].

7.4.4 Verifying Energy and Force Calculations with LAMMPS

Many of the calculations depend on the energies computed as well as accurate computation of the forces. LAMMPS is a widely used molecular dynamics code and many potentials already exists in the format required for LAMMPS. The energies and forces computed by the EAMPA code were compared for a number of configurations and potentials against the energies and forces computed by LAMMPS to validate the code developed here. The results between the two codes agreed.

7.5 Application of the EAMPA Code

7.5.1 Investigating the Transferability of Existing Potentials

There are currently a number of EAM Iron potentials, as well as a MEAM potential for Fe-Pd. Several pure element potentials also exist for Ruthenium and Palladium. Whilst it would be convenient to have a one size fits all potential, this is not the case. Taking the Ruthenium potential as an example, in work that required the reproduction of stacking fault and surface energy of the element, the existing potentials were inadequate and a new potential was developed[205].

Several potentials were selected to test transferability. First, the FCC Sheng potential for Aluminium was applied to both FCC and BCC structures in the EAMPA code. The properties of FCC Aluminium are known, and those for BCC Aluminium were calculated using DFT. This investigation shows that some properties, such

as the lattice constant, does transfer quite well to the new structure, but the bulk modulus and C_{11} elastic constant are both poorly reproduced (table 7.15).

Property	Al FCC Exp.	Al FCC Sheng	Al BCC DFT	Al BCC Sheng
a0	4.05	4.02	3.23	3.18
e0	-3.36	-3.36	-3.26	-3.27
b0	76	86	69	50
C_{11}	114	124 (125)	47 (47)	11 (5)
C_{12}	62	67 (66)	81 (24)	70 (63)
C_{44}	32	29 (32)	40 (12)	29 (42)

Table 7.15: Applying the Sheng Al potential, developed for FCC crystals, to the BCC structure. Elastic constants are calculated using the method by Mehl et al with values from the method by Ravindran et al in brackets.

The Mendelev 2003 potential was developed to fit the properties of BCC Iron, but also to reproduce the transition from BCC to FCC at high temperatures. However, when using this potential for low temperature FCC Iron, it does not reproduce the properties predicted by DFT. All 9 orthorhombic elastic constants have been omitted in table 7.16, but there is a significant difference for the three elastic constants given as well as the bulk modulus.

Property	Fe BCC Exp.	Fe BCC Mendelev	Fe FCC DFT	Fe FCC Mendelev
a0	2.86	2.86	3.42	3.59
e0	-4.32	-4.13	-4.26	-4.01
b0	170	176	226	98
C_{11}	243	255 (260)	364	6 (79)
C_{12}	145	137 (143)	142	145 (69)
C_{44}	116	114 (111)	186	86 (48)

Table 7.16: Applying the Mendelev 2003 potential[143], developed for BCC crystals, to the FCC structure. Elastic constants are calculated using the method by Mehl et al with values from the method by Ravindran et al in brackets.

Finally the Ackland 1997 potential for Iron was tested. This older potential gives a better value for the cohesive energy when transferred to the FCC structure, but other properties such as the relaxed lattice constant are quite different (table 7.17).

Property	Fe BCC Exp.	Fe BCC Ackland	Fe FCC DFT	Fe FCC Ackland
a0	2.86	2.87	3.42	3.62
e0	-4.32	-4.32	-4.26	-4.26
b0	170	200	226	141
C_{11}	243	265 (248)	364	117 (140)
C_{12}	145	168 (151)	142	153 (117)
C_{44}	116	125 (128)	186	164 (94)

Table 7.17: Applying the Ackland 1997 potential[222], developed for BCC crystals, to the FCC structure. Elastic constants are calculated using the method by Mehl et al with values from the method by Ravindran et al in brackets.

This investigation into existing potentials shows that whilst some of the properties are replicated, many do not match the DFT computed values. This supports the derivation of new potentials to be fit that replicate the DFT values and take into account the characteristics of a number of provided configurations. Additional plots are included in the appendix that show the expected equations of state compared to those generated using the above potentials (section P).

7.5.2 Fitting Potentials with EAMPA

The fitting process was computationally intensive and utilised nodes on the University of Birmingham Blue-BEAR computer. At the time of writing, the code was able to fit energy, forces and bulk properties. Whether the method with which the stress is calculated in PWscf was the same as that used in the EAMPA code was unclear. As a result it is suggested to set the weighting of stress to zero.

Choice of Function and Functional

A number of functions and functionals have been used previously and these are discussed in detail in chapter 5. Several were used to attempt to fit the potentials, but eventually the cubic splines and tight binding type functional, as used by Mendelev, Ackland, Bonny et al, were selected.

Initially, the computer program DIRAC was used to generate the density function, as it computes the radial electron density distribution, as seen in fig. 7.11 and fig. 7.12. Unfortunately, as may be expected, there were a number of issues in using the density distribution of a single atom. The atomic separation for both Fe and Pd, for a relaxed crystal, begins at approximately 2-2.5 angstrom, but most of the electrons, from all shells, are distributed much closer. The plot does not take into consideration the valence electrons shared in the crystal, bonding the metal atoms together.

From the point of view of the potential, the pair potential will dominate at small separations due to the ZBL function. The density function will be important at separations of 2 angstrom up to the cutoff radius (5-6 angstrom). For this reason, in addition to moving away from attempting to represent the underlying physics, the idea to use the DIRAC computed densities was abandoned in favour of a cubic spline.

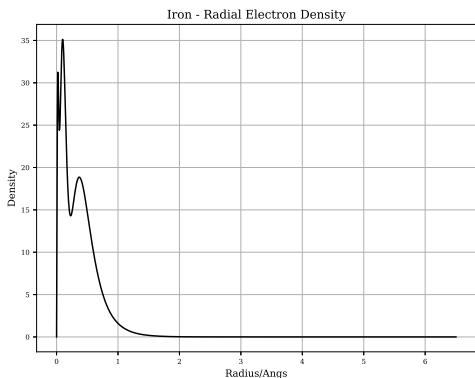


Figure 7.11: DIRAC calculated radial electron density for Iron

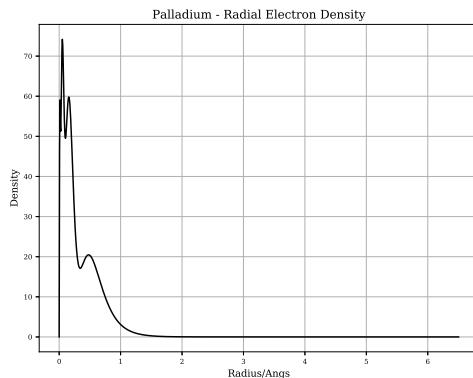


Figure 7.12: DIRAC calculated radial electron density for Palladium

One interesting feature that Bonny used in his code was to add a hard core spline to the pair potential with a positive coefficient. This was to ensure that at small separations, say below 1.5-1.8 angstrom, the atoms would repel. The cubic spline used in this work for the density also has a parameter as a positive coefficient in order to stop negative densities for small separations. This is dependent on the number of cubic splines used, and is a feature that can be turned off in the EAMPA code.

A cubic spline was also used for the pair potential, and as with Bonny's code, this had a hard core potential but in addition to this a ZBL core was added to the entire potential to ensure the exponential increase in energy as modelled with the ZBL function. This was to ensure the potential would both model the material when out of a radiation field, where the atoms are several angstrom apart, but also able to capture a damage cascade where the separation drops and the ZBL becomes the dominant feature, over both the pair cubic spline and the embedding functional.

There was a choice of several embedding functionals but one based on the Finnis-Sinclair and EAM was used. It consisted of the square root term but also a zeroth power and square term for greater flexibility. The addition of a fourth power term had been trailed, but occasionally when fitting the parameters the coefficient of the fourth power would become negative leading to a negative energy at high electron densities.

Searching Parameter Space

Improving the potential was an iterative process that required multiple runs of EAMPA feeding the last potential as a starting point for the next potential. In the first instance, the exact bulk property fitting code of Bonny was used, but it did not always find the parameters that reproduced the data points it was given. This was perhaps due to a lack of data and knowledge on how to use the code effectively, but it did give a starting point with a modest number of cubic splines for the density and pair functions.

These initial set of parameters were then used with EAMPA. The first task with this code was to find parameters that replicated the energies, forces and bulk properties for each of the pure elements.

The type of search could be adjusted for each fitting job, but the typical process was as follows. A short random search through parameter space followed by a small simulated annealing search with a wide search area. This was then followed by a genetic algorithm search with quite a wide search area. Next a second genetic algorithm search over a smaller range of parameters but with more iterations and a larger population, and this step used most of the computation time. Finally a short simulated annealing and local BFGS minimizer were used before outputting optimised parameters. A sample input is given in listing 7.1, but the user is free to add fewer or more steps and search types.

Listing 7.1: Parameter search settings when fitting in EAMPA

```

1 #FIT type=RAND niter=1000
2 #FIT type=SA niter=1000 titer=10 tstart=100.0 tend=0.001 pfact=0.5 pvar=10.0 vartype=add gaussian=True
3 #FIT type=GA niter=200 popsize=500 fresh=0.2 search=w
4 #FIT type=GA niter=5000 popsize=2000 fresh=0.2 search=n
5 #FIT type=SA niter=1000 titer=15 tstart=1.0 tend=0.001 pfact=0.5 pvar=0.01 vartype=add gaussian=True
6 #FIT type=BFGS

```

The code has an option to set the random seed such that every job with that seed and settings will return the same output. This allowed ten or more jobs to be run in parallel, with 4 cores per job, and a different seed for each job. Typical run times were 3-5 days per iteration and several iterations required.

At the start of the fitting process all available configurations were used, but it turned out to be difficult for the code to satisfy all the constraints given the simplicity of the EAM potentials. Towards the end of the process many of the configurations were removed with bulk and surface/slab configurations being left. This was in order to train the potential specifically to these configurations in favour of the entire set.

Once the pure potentials for Fe, Pd and Ru were determined, they each underwent an effective gauge transform, with the transform being that used by Bonny et al[139][146]. Due to this transform, they were converted from analytic potentials to numeric.

The fixed numeric functions were then used in the binary alloy fitting where the only parameters adjusted are those for the cross pair potential. The searching method was similar to that used for the pure elements, but the configurations were changed to those containing both species of atom. The pure element configurations were removed as these potential functions were now fixed. The bulk property fitting was also turned off as this fitting was only applicable to the pure elements and had no affect on the cross species potential.

7.5.3 Iron-Palladium and Iron-Ruthenium Potentials

Reference Database

The database of reference configurations was used to derive this potential. It was designed for FCC Fe rather than BCC Fe, and this is one point that distinguishes this potential from those that already exist. As discussed in section 7.5.1 a potential developed for one application isn't necessarily transferable to another.

There were a number of options available when choosing certain parameters, such as using DFT to compute cohesive energies based on relaxed and isolated configurations, or to extrapolate based on known energies. The choices made are listed below and these apply to both the Fe-Pd and Fe-Ru potential.

- all cohesive energies were computed based on known experimental values and the DFT computed relaxed energy of those structures
- all bulk property values are those computed using DFT

During the fitting process, there was a desire to have a reasonably good fit to bulk property values as well as the defect configurations, perturbed atom configurations and so on. Weights were applied to the bulk properties as a whole and individually to the property, such as lattice parameter, cohesive energy and so on. They were also applied to the reference configurations as a whole and to individual configurations.

The potentials for Pd and Ru were fit only to their FCC structures and more weight was given to the lattice parameter and cohesive energy. As the DFT computations had already been carried out for BCC Fe, these were also used to fit the potential for Fe, but a much lower weighting was given to these than the FCC structure. Again, more weight was given to the lattice parameter and cohesive energy than for other properties.

There were many configurations available, but given the reason for the potential is to model PGMs on or near the surface of a grain of steel, more weight was given to the bulk and slab/surface configurations than defects and other distorted configurations.

Potentials

The full potentials are available from https://github.com/BenPalmer1983/fepd_feru_potential and they are ready to use with LAMMPS. The functions for pure Fe are the same for both potentials and give the same bulk properties, but the binary potentials and of course those for pure Ru and Pd differ from one another.

A summary of the properties is given in table 7.18. This data and more are available in appendix U.

Both the Birch-Murnaghan and Rose-Vinet equation of states were used to fit the potentials, and these are useful for fitting the bulk modulus (Birch-Murnaghan) and behaviour of the potential over a wider range of values for the lattice parameter (Rose-Vinet). The Rose-Vinet fit for Fe is given in fig. 7.13 for both the FCC and BCC structures.

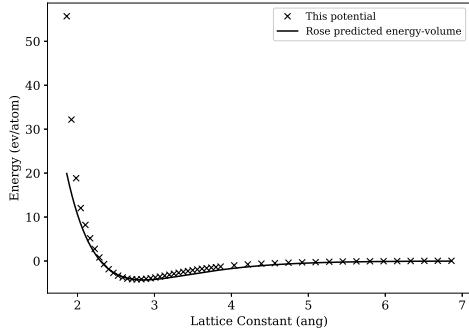
The Rose-Vinet fit for both Pd and Ru is given in fig. 7.14 for just FCC structures. The potentials were then used to calculate the energies predicted by DFT and how well these fit, for both potentials, are given in fig. 7.15.

Configurations of atoms transitioning from bulk to a slab were used in fitting the potentials. The comparison between DFT and the derived potentials are plotted in fig. 7.16. There is a good match in the DFT and potential energies for Fe and Pd, and a discrepancy of less than 0.1eV per atom for Ru.

Property	Fe FCC				Fe BCC		Pd FCC		Ru FCC	
	DFT	Potential	Mendelev[143]	Ackland[222]	DFT	Potential	DFT	Potential	DFT	Potential
a_0 (angs)	3.42	3.42	3.59	3.62	2.80	2.76	3.93	3.89	3.81	3.79
x	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
y	1.054	1.05	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
z	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
e_0 (eV)	-4.26	-4.27	-4.01	-4.26	-4.32	-4.16	-3.91	-3.78	-6.62	-6.61
B_0 (a) (GPa)	226.1	246.6	98.4	141.0	239.2	229.0	184.4	165.0	307.8	331.1
B_0 (b) (GPa)	222.0	235.2	88.5	135.5	205.3	207.6	173.7	156.0	303.3	320.4
C_{11} (GPa)	364.6	312.6	79.2	140.5	250.1	190.1	218.5	191.1	471.5	375.0
C_{12} (GPa)	141.6	182.3	68.6	117.5	182.8	248.5	151.4	152.0	219.1	309.2
C_{13} (GPa)	233.8	222.8	105.7	130.9	-	-	-	-	-	-
C_{22} (GPa)	298.7	321.6	161.7	228.5	-	-	-	-	-	-
C_{23} (GPa)	130.4	182.3	68.6	117.5	-	-	-	-	-	-
C_{33} (GPa)	364.6	312.6	79.2	140.5	-	-	-	-	-	-
C_{44} (GPa)	186.3	171.9	47.8	93.8	139.4	168.7	80.3	58.8	245.0	97.5
C_{55} (GPa)	266.8	223.1	73.7	108.9	-	-	-	-	-	-
C_{66} (GPa)	186.3	171.9	47.79	93.8	-	-	-	-	-	-

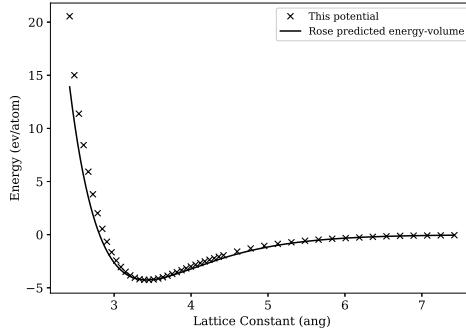
Table 7.18: A summary of bulk properties from these potentials. The bulk modulus is computed from the Birch-Murnaghan equation of state (a) and an average of the Reuss and Voight bulk modulus (b). The FCC properties of Fe are compared to those predicted with two existing potentials designed for BCC Fe to highlight the issue of transferability and the need for new potentials.

Rose Equation of State



(a) BCC Iron

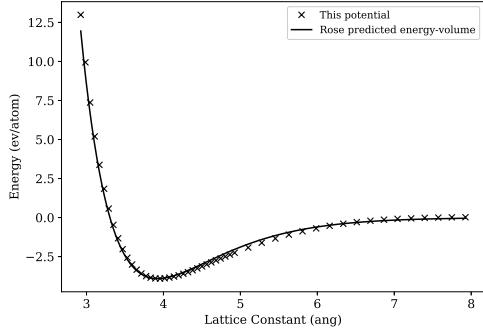
Rose Equation of State



(b) FCC Iron

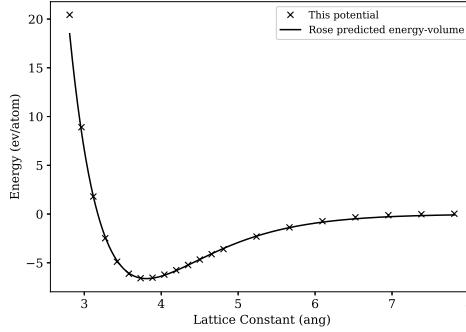
Figure 7.13: Rose-Vinet plots for Iron

Rose Equation of State



(a) FCC Palladium

Rose Equation of State



(b) FCC Ruthenium

Figure 7.14: Rose-Vinet plots for Palladium and Ruthenium

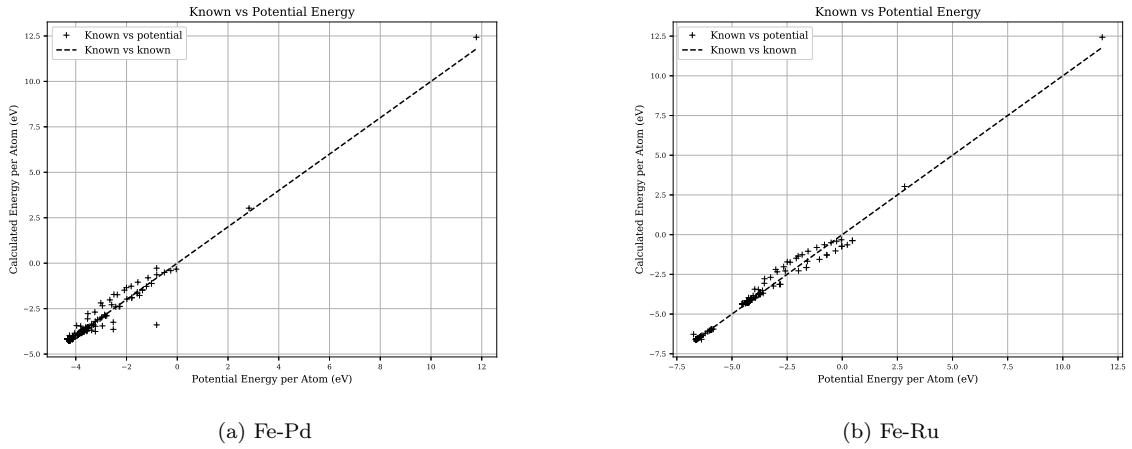


Figure 7.15: Plotting how well the potentials reproduce DFT energy values

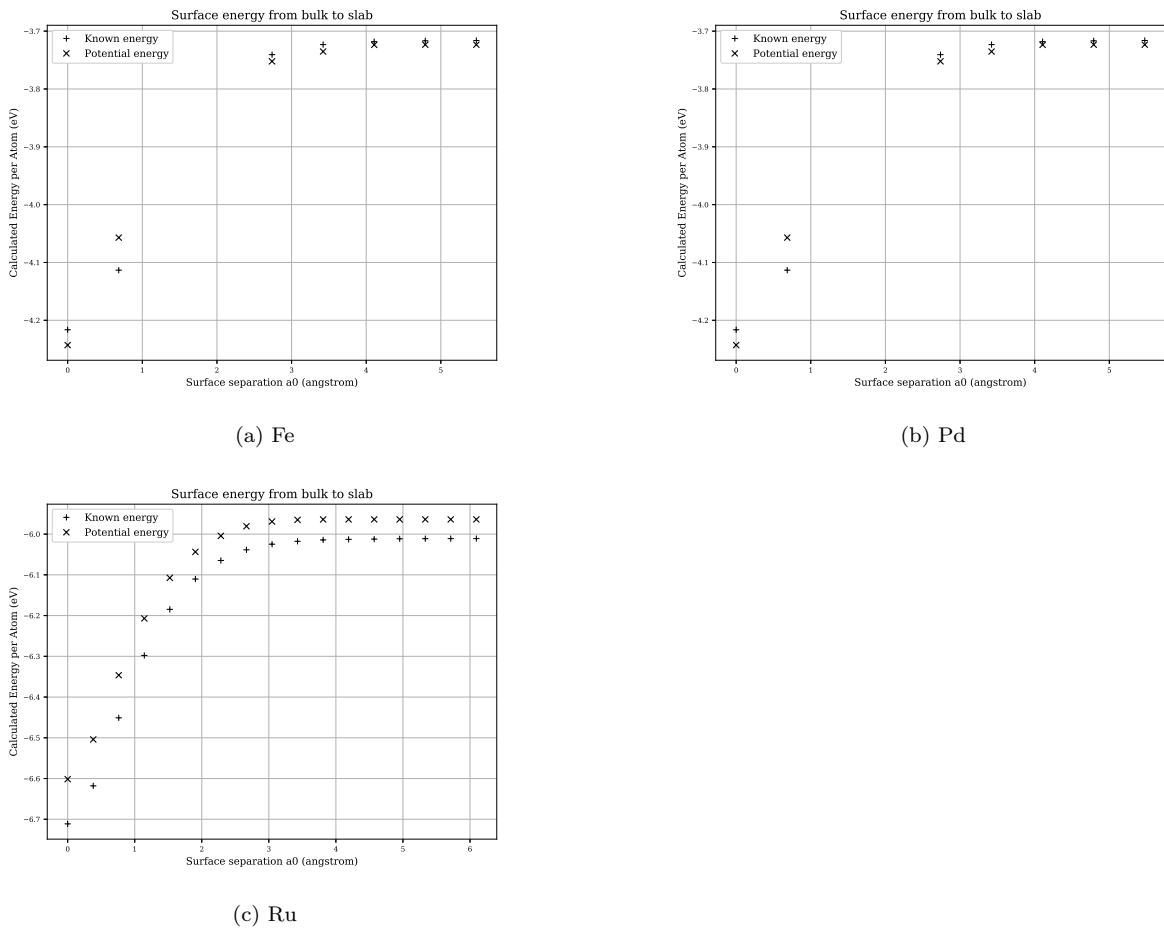


Figure 7.16: Energy plotted as both a slab and two surfaces begin to form. The plots give the energies predicted by DFT and computed using the potentials from this work.

7.6 DL_POLY Contribution

DL_POLY is a Molecular Dynamics code developed by W. Smith, T.R. Forester and I.T. Todorov at Daresbury Laboratory in Warrington. It is written in Fortran and, before the modifications, included a number of potential types for metals and several of these are listed below.

- Finnis Sinclair
- EAM
- EEAM

The Finnis Sinclair is a particular form of the EAM potential, and EEAM is a modification where, if the metal is an alloy, the density and embedding functional for each atom type are treated separately.

A meeting was held with Dr. Todorov at Daresbury Laboratory, where a brief overview of the relevant code was given. After this point, and corresponding by email, the two band EAM type was added (2BEAM). The code was altered with the addition of an array used to store the d-band density function and d-band embedding functional, with the s-band function/functional using the original density function and embedding functional arrays. The calculation of energy and forces on atoms was also altered to make use of these arrays where a two band embedded-atom method potential is used.

Listing 7.2: DL_POLY 4.05 mailshot extract

```

1 DL_POLY_4.05: New Release & Events - MAILSHOT 013
2
3 NEW FEATURES \& IMPROVEMENTS
4 -----
5 1. New two band (2B) EAM and EEAM potentials for metals (TEST45 and TEST46).
6
7 Acknowledgements
8 -----
9 Ben Palmer \@ University of Birmingham (UK) for contributing to the
10 development and testing of the 2BEAM for metals;
11
12 Regards,
13
14 Illian Todorov
15 July 2013

```

Extracts of the modified DL_POLY code are included in appendix X. Dr Todorov's assistance in correcting the modifications and integrating them into the release version of the code was greatly appreciated.

Chapter 8

Conclusions

A large enough addition of Cr to steel gives protection to corrosion through a passive protective layer. This is depleted during irradiation, but the addition of PGMs offers alternative protection providing they are not depleted as well. A loss of protection leads to IGSCC.

Ion irradiation provides an alternative method to neutron irradiation. Equations and a code have been developed in this work to predict how radioactive ion irradiated targets become, and allows computing radioactivity at given ion energies and DPA.

Interatomic potentials have been derived for Fe-Pd and Fe-Ru binary alloys in the FCC structure. They have had limited testing but are in LAMMPS format and are available to use. This is a step towards the goal of modelling the grain boundary with MD or akMC.

8.1 Inter Granular Stress Corrosion Cracking

Austenitic stainless steel is particularly susceptible to inter granular stress corrosion cracking. It has good properties, including a good resistance to corrosion due to its high Cr and Ni content. Under irradiation, the Cr at grain boundaries is depleted with the formation of Cr carbides. There is also segregation due to the inverse Kirkendall effect, driven by the radiation, that preferentially depletes Cr leaving Fe and Ni. Three requirements for inter granular stress corrosion cracking to occur are:

- a susceptible material (austenitic stainless steel)
- stress, welds, swelling, pressure, any applied stress (high pressure in reactor environment, swelling due to irradiation)
- a corrosive environment (radiolysis of water)

The passive Cr layer, under normal conditions, protects the steel from corrosion. The steel is made up from small grains, and the surface of these grains of crystalline metal are where the protective layer covers. As the percentage of Cr drops below the threshold required for formation of the passive protective layer the surface becomes prone to corrosion.

Small quantities of PGMs may be added to a steel to also increase its corrosion resistance. PGMs are rare and much more costly than Fe, Ni, Cr. Where irradiation is not present, it has been shown experimentally that the benefit due to Pd is lost due to the formation of Pd-Mn nanoparticles, where Mn is present in the steel, although no such particles are present when Pd was replaced by Ru. Pd does provide corrosion protection via

cathodic modification when Mn is reduced to very small amounts as it no longer forms Pd-Mn precipitates during sensitisation. The outstanding question is whether or not Pd or Ru are depleted, enriched or are unaffected at the grain boundary when under irradiation.

8.2 Predicting Ion Induced Radioactivity

Testing a material inside a nuclear reactor is an expensive experiment. There are many more ion sources around the world than high flux neutron sources, and ion beams are easier to direct due to the charge of the ion versus the neutral charge of the neutron. The ion energies required to create similar sized damage cascades in the material are high enough to also have a chance of transmuting the target nuclei. These transmuted nuclei are highly likely to be left in an excited state and will either be metastable or radioactive. To reach damage dose levels comparable to a component in a Gen IV reactor by the end of its life (up to 200 DPA), the target material will become dangerously radioactive.

Bateman's equation may be used to calculate the amount of an isotope in a decay chain of several isotopes, and thus how radioactive an isotope would be after a certain time. Major modifications were required to be made to the equation to also include branching factors and source rates for isotopes generated by an ion beam.

8.2.1 Modified Equation

Whilst the equation was first tackled by solving the differential equations head on, it became obvious that using Laplace transforms would be the logical choice; this was also the route Bateman followed when deriving the original equations. Once transformed, a numerical algorithm (Gaver-Stehfest) was programmed and used to solve, with some success.

The numeric method is not an exact solution, and the errors incurred often resulted in negative amounts of an isotope, and this was unacceptable. More progress was made towards solving the problem analytically, and this was achieved by using partial fractions, obviating the need for a numerical method.

The analytic solution was tested against a simple numeric solver and the results were in good agreement between the modified equation and numeric solutions (appendix B.3.1).

8.2.2 Activity Code

The Activity code was created to automate the process of calculating the activity of an ion irradiated target. The program:

- reads in simulation settings (target composition, thickness, duration etc)
- reads in ion cross section data from the Talys database
- processes SRIM exyz data points
- calculates the amount of each isotope by the end of the simulation
- calculates the activity of each isotope and the dose that an average human would be exposed to

A number of simplifications were made to the model but it predicted the radioactivity of certain peaks reasonably well with a sample of proton irradiated pure Fe. Given more time a range of alloys would also be irradiated by a proton beam and, once their activity is measured at several time intervals after irradiation, they would be compared to the activity predicted by the Activity code.

The first version of the Activity code was replaced by a second version that used Python rather than Fortran. The first version was more cumbersome for new users and it crashed for certain isotope, and these problems were fixed in the new version.

8.2.3 Iron Irradiation

During the experimental stage of this work it was determined that the isotope to focus on would be $^{55}_{27}Co$ and the associated 931keV peak it generates would be the peak to measure. The equipment was calibrated specifically for the purpose of measuring this gamma. If this is the case then the value predicted by the Activity V2 computer code is in good agreement (table 6.6).

The Activity code collates and displays data generated from the cross section database and JEFF 3.3 decay data file. It is clear that there should also be a nearby 935KeV gamma from the decay of $^{52}_{25}Mn$. The sum of these nearby gammas is predicted to be almost 120kBq and, if both peaks were tallied in the experimental work due to the energy resolution of the detector, the predicted value is three times that of the measured value of 44kBq.

It is quite possible that the code is giving a value three times larger than expected and reasons for this will be discussed below.

8.2.4 Molybdenum Irradiation

Similar to the results discussed for Fe, the activity results computed for Mo have positive and negative attributes. There are gaps in the available cross section data, but the most prominent gammas are predicted by the computer code. The radioactivity computed is too high, but for each of the five peaks measured by experiment, the values are out by a similar factor of approximately 7-9, depending on the peak.

8.2.5 Beam Flux Reduction and Gamma Attenuation

The Talys derived cross section data used by the Activity code has been checked for a number of cases against EXFOR data and is in good agreement with this. Certain data for reactions is missing in particular those of observationally stable and isotopes with extremely long lived isotopes, but this would have a minimal impact due to how low the radioactivity would be.

Thin foils are ideal to compare experimental results to the data files and activity code as ions will not lose a great deal of energy travelling though such a shallow target. However, the much thicker targets used here for Fe and Mo reduce ion energies significantly, if not completely.

It could be argued that as ions pass through the target the energy and intensity of the beam decreases, and this is not currently modelled with the activity code. By imagining the target as thin laminae perpendicular to the beam it is easy to consider that the ions will enter the first layer and lose those that react with target atoms. As the beam enters the next lamina it will have a lower energy, either due to electronic stopping or deflection by nuclear collisions, and a lower intensity as a proportion of the protons will be absorbed.

The change in energy is accounted for in the code through the SRIM exyz ion trajectory file but how the beam intensity decreases is not. It is important to approximate how many ions are captured through nuclear reactions and whether it is a significant percentage of the total number of ions. Particles and residuals were calculated to be produced at a rate of approximately 6×10^{10} per second for the Fe target and 7×10^{10} per second for the thick Mo target. The beam rates were 3×10^{12} and 3×10^{13} protons per second respectively and so, even if the estimate that a proton was removed for each reaction the beams would only lose 2% and 0.3% of their intensity due to transmutation reactions.

Another possibility for the over estimate in prediction is the attenuation of gammas travelling through the target material to the detector. This was discussed in the work of my colleague[200] and the inhomogeneous distribution of radioactive material in the Mo target was discussed in this work (section 6.7.2). Calculations using the NIST attenuation coefficient for Mo show that 500keV gammas have their intensity reduced to 91% through a 1.0cm target. The intensity of 1MeV gammas is reduced to just 94% through the same target.

The reduction through loss of ions due to reactions and attenuation of gammas does not explain the increase in predicted activity. There may be another reason behind this, or there may be issues with the experimental work.

8.2.6 More Experimental Data Required

I have not had a great deal of experience practically measuring gammas, and I would expect there to be errors introduced due to my own inexperience. The range of beam energies, elements and beam duration are also very restricted.

In past experimental work discussed in section 4.4.3, thin foils were used to measure reaction probabilities, and this data influences models such as TALYS. In some two out of the three examples reviewed, the data beam energy was degraded using a stack of foils perpendicular to the beam, with those closest to the aperture receiving the highest energy protons. It is hard to imagine whether foils stacked in this way would be much different to a solid target of the same thickness, but this should be tested experimentally.

Ideally a range of target materials of varying thicknesses would be irradiated with protons of energies ranging up to the limit of the cyclotron, perhaps 60MeV, and the Gamma spectra of each measured at set times after irradiation. This data would be compared to the gamma lines predicted by the Activity code to help give a better understanding of the accuracy of the code.

8.2.7 100DPA Activity Predictions

Using dose rates calculated with SRIM, the Activity code was used to predict how radioactive Fe would become at a dose rate of 100DPA. It assumes the beam source is the Scanditronix MC40 using protons at maximum current for an uninterrupted duration of 134-391 days. In the worst case scenario, 25MeV protons result in over 20 Gys per hour if irradiated to the required damage dose, and this level of dose would be extremely dangerous and probably fatal if exposed to for half an hour. However, by reducing the beam energy to 15MeV radioactivity of the target is reduced by a factor of 1,000. It may be reduced further by lowering the energy to 5MeV or 10MeV, but for a 0.5mm piece of steel at these energies the ions would not have enough energy to pass through and the damage would be disproportionately concentrated in the top 100-250 microns of the target.

8.2.8 Original Contribution

The modified Bateman equation as derived is an original contribution to knowledge. The Activity code is also an original contribution, and its use shows how adjusting beam parameters will reduce the amount of radioactive material created and the risk to those handling irradiated samples. It is also a useful tool to highlight problematic isotopes and reactions of interest within a complicated material.

8.3 Derived Potentials

8.3.1 Restricting Potentials to Binary Alloys

It would be too complex a task to develop an interatomic potential that includes Fe, Pd, Ru, Cr, Ni and any other elements that make up a typical sample of austenitic stainless steel. The binary alloy potentials for Fe-Pd and Fe-Ru are a starting point to with the possibility of adding more elements to the potential in the future.

Austenitic steels have a FCC structure and this is why the potentials were trained with a higher weighting to FCC configurations. As only Pd exists under normal conditions in this state, the properties for all three elements were computed using DFT.

8.3.2 Collinear Spin DFT

With most models, there will be a trade off between simplicity, accuracy and the computational cost of the model. The parameters that had the largest impact on how long a DFT calculation will take to compute are the planewave energy cutoff and number of k-points used to integrate the Brillouin zone. These should be reduced as much as possible, while keeping the energy and force results within a tolerance determined by the user.

A choice in the complexity of the calculation also had to be decided upon, and this was whether to use a non-spin, collinear spin or non-collinear spin calculation. Examples of room temperature ferromagnetic and antiferromagnetic materials, Fe and Cr respectively, were investigated with DFT using both non-spin and collinear spin. These calculations showed that when magnetism is ignored the optimum structure for Fe is FCC, but when the spin of unpaired electrons is accounted for DFT then correctly predicts the optimum structure as ferromagnetic BCC. Similarly when spin polarized DFT calculations are performed for Cr the optimum state is shown to be antiferromagnetic BCC, as expected. It was therefore deemed necessary to use these more computationally intensive calculations.

Non-collinear spin was not considered as it would increase the resources required further and would possibly require the testing of suitable pseudopotentials. The collinear calculations were adequate for modelling bulk Fe, but there could be merit in using non-collinear spin calculations in the future due to the local effects individual doped PGM atoms may have plus investigating the effect of a surface on the alignment of atoms nearby.

8.3.3 Automated Convergence Calculations

A Python code (QECONVERGE) was developed to automatically converge the aforementioned parameters. Whilst automation worked well for the planewave cutoff parameters the choice of smearing type, the amount of smearing and number k-points was more difficult to automate. It was more useful to select a range of k-point values and smearing values and produce 2D force and energy plots so the user could make a decision based on these and choose the parameters that best suit their requirements.

During testing, a much larger number of k-points were needed for the predicted properties of Aluminium to replicate the known properties reasonably well. Unfortunately, the number of k-points used for Fe and Pd was constrained by the amount of time and the RAM available on the compute nodes. Ideally, more k-points would have been used, but this was not possible.

8.3.4 Bulk Property Calculations

As discussed in the previous section, spin polarised DFT were used for all first principles calculations in this work. Although a smaller number of k-points were used than would have been desired, the calculations required to compute these properties continued for more than a month.

Rather than calculate the input files manually, a program was created to automate this process (QEEOS). It was designed to cache input and output files for the DFT program, PWscf, such that it would load an output file from the cache if it had already successfully run, to reduce the calculation time.

In the case of Fe, the relaxed (energy minimised) shape of the antiferromagnetic FCC Fe was not cubic, but orthorhombic. The equation of state calculation is specifically for a cubic crystal, but the elastic constants were calculated for the orthorhombic crystal. From the 9 calculated elastic constants the melting point, bulk modulus, shear modulus and Young's modulus were calculated (section 7.3.4 and appendix N.5). The resulting structure was stable and were well within a magnitude of the known values for BCC Fe. These data are an original contribution to science. The properties were also computed for Pd and Ru and were used in the derivation of interatomic potentials.

8.3.5 Iron-Palladium and Iron-Ruthenium Configurations as Fitting Data

To be able to fit the Fe-Pd and Fe-Ru potentials, the bulk properties were computed and a range of other configurations were generated. A number had atoms randomly perturbed from their perfect positions, others represented slabs, defects and so on.

The energy and force data were entirely from first-principles calculations, and ideally there would have been more configurations and computed with a higher number of k-points as well as a reduction in the smearing value, if possible. In addition, a wider range of vacancies, interstitials and mixing concentrations would have been desirable. Unfortunately many of the configurations failed to converge, and this was particularly prevalent where the atom configurations were far from the perfect crystal, and even more so when multiple elements were used in a calculation.

The atomic percentage on Pd or Ru doped is half a percent which would be better represented with a one PGM atom in a 4x4x4 FCC 256 atom supercell rather than the 32 atom cells used, but again this was a constraint imposed by computational resources and this lead to the majority of the calculations being performed on 32 atom supercells, with relatively few 256 atom supercells.

8.3.6 Iron-Palladium and Iron-Ruthenium Potentials

Both potentials have had minimal testing using the molecular dynamics package LAMMPS. In each case LAMMPS was able to read in the potential and run a simulation with a Nose-Hoover thermostat. When given a relaxed start point, the simulations were well behaved either as a nano cube particle surrounded by a vacuum or as the bulk material represented by a cube subject to PBCs.

As the relaxed structure from DFT calculations for Fe was FCT, the structure is not symmetric and a directional potential would better represent the increased lattice parameter in the y axis (figure 7.4). One possibility would be to replace the EAM potential with a MEAM, but this is discussed further in section 9.2.4.

The individual potentials reproduce the bulk properties, as calculated by DFT, reasonably well with the lattice parameters and cohesive energies being within a few percent of the experimental values. The poorest properties in the fitting process were the elastic constants, in particular the C_{44} value for Ru. These could be improved by running the code for a longer time period.

The Rose-Vinet equation of state for each pure element fits well and takes on a shape that would be expected, with an increase in energy where the lattice parameter is reduced, a minimum where the relaxed lattice parameter is and a reduction in energy to zero as the atoms are pulled further and further apart. Where the creation of a slab with two surfaces is modelled, the predicted energies for each pure element match well with the DFT energies. Out of the three elements, Ru fits least well as far as the formation of a slab is concerned with a consistent energy difference of at least 0.05eV.

Overall, both of the potentials reproduce the energies calculated using DFT well across a range of configurations. There is definitely room to improve and build upon these potentials given more time.

8.3.7 Original Contribution

The fitting code is an original contribution as it allows the fitting of metal potentials in the form of the embedded-atom method and two band embedded-atom method to DFT data and bulk properties. Other similar codes do exist, such as FitPot and PotFit, but at the time of developing the EAMPA code, PotFit did not have the ability to fit analytic potentials (although, this feature is now available) and does not have, in its available form, the ability to fit the potential to the bulk properties of the material (although the Sheng version did have this ability). The code developed here may be modified in the future to add more features and properties to fit potentials to. The two potentials developed are also an original contribution due to the issues in transferability of existing potentials.

8.4 Two-Band EAM Contribution to DL-Poly

During the early stages of this work it was clear that the Embedded Atom Method was a good candidate for the type of potential to be derived as it was well suited to modelling metals and had been used previously (along with the similar FS potential) to model radiation damage[31]. Work by Prof. Ackland at the University of Edinburgh considered using a separate density function and embedding functional to represent the S-Band and D-Band of transition elements such as Caesium. Work by P. Olsson and J. Wallenius also used this two band version to model Fe-Cr with a later Fe-Cr potential being developed by Bonny et al to replicate the mixing enthalpy as a function of Cr content.

After meeting with Prof. Todorov at Daresbury Laboratory, the source code for DL-Poly was modified to include new keywords, additional arrays to store the two-band density and embedding data for the second band and modifications to the energy and force subroutines. This was then released with version 4.05 of DL-POLY in July 2013.

Ultimately, only EAM potentials were fit in this work. In the future, as more elements are added to the potential, in particular Cr, the 2BEAM may be required for the reasons discussed in section 5.3.3. The molecular dynamics code and fitting code both have the capability to use this type of potential.

Chapter 9

Future Work

There are a number of improvements that could be made to the Activity code, but the suggestions need to be evaluated in terms of their benefits and computation time. More experimental data is also required. As for the potentials developed here, more time might be invested in both DFT calculations and fitting time. Following this, the potentials must be evaluated using LAMMPS or DL_POLY before being used to model grain boundaries.

9.1 Activity Code

9.1.1 Experimental Activity Readings for Ion Irradiated Targets

The Activity code was developed using data generated by the Talys nuclear reaction code. To test the validity of the Activity code, an Fe sample was irradiated and its activity was measured several days after irradiation.

To validate the code further, a range of pure targets and alloys would be irradiated by a cyclotron, with targets of varying thickness and beams of varying fluences and energies. This would generate data to be used to compare to the results predicted by the code. This would require significant resources in terms of beam line time on the cyclotron as well as an experienced experimental physicist.

9.1.2 Code Improvements

A number of improvements could be made to the code. While Python is a popular language, and a feature rich one, there are still benefits in using a language such as Fortran. This is especially true when coupled with OpenMP. There certainly are parts of the code that could be improved by using F2PY to decrease runtime.

All versions of the code have relied on data from SRIM. It might be useful to integrate or build an ion transport class so a calculation may be run once, rather than having to first generate the transport data on one platform and run the Activity code on another.

It should also be investigated whether or not a history of many particles is even needed at all as there already exists average energy stopping tables. If there is little difference in calculated activity when using the histories of a thousand ions and energy stopping tables, then the stopping tables should be used instead. It would remove the need to generate the ion history, which in itself is a long process, and it would speed up the computation of target activity.

The energy range of cross section values and particle types could be extended to higher energies and a wider range of particles including deuterons, tritons and helium ions. This would require more time but would be

useful in the case of the University of Birmingham cyclotron as it is capable of accelerating these particles as well as protons.

Currently the code does not take into account 511KeV gammas created by the annihilation of positrons from β^+ decay or pair production from higher energy photons. This will affect the calculations during irradiation and during the cooling period. Where the transmuted nuclei is left in an excited state due to the energy difference, the gammas that will be released are not yet accounted for. This is more important during irradiation and may not need to be considered providing there is adequate shielding whilst the target is in the ion beam.

As the target becomes thicker, the orientation of the irradiated target to the end user expected to handle the sample will become more important. Incorporating this into the end calculation to display the gamma fluence radially in 2 dimensions from the target would help the user position themselves relative to the irradiated face of the target to minimise the dose received.

Whilst the target is being irradiated, there will be a high number of gammas as well as other radiation such as neutrons, protons, electrons and other light ions. It has always been assumed that this process will be carried out behind adequate shielding as it is only a danger when the proton beam is on. When out of the beam the primary concern to a person nearby the target are the gammas released by radioactive decay. There may also be reasons to consider the amount of beta and alpha radiation emitted by the target, although this might only be a concern if the target is to be machined, releasing ingestible radioactive dust, or handled without protection where the beta particles will be in a position to travel through skin. Ideally, both these circumstances would be avoided, but the utility could be added to the code.

More features could be added including a choice of dose units and whether or not the dose is calculated for a given volume of space or for a given person with certain parameters.

9.2 Potential Fitting and MD Simulations

9.2.1 Larger Supercells

In many examples in the literature, the supercell sizes were at least 4x4x4 containing 256 atoms for FCC crystals. These require much more memory and processing power, especially considering the lack of symmetry and inclusion of magnetism in the calculations.

If these potentials were to be improved in the future, all the calculations would contain approximately 128 atoms for BCC and HCP configurations and 256 atoms for FCC as all the configurations would be based on 4x4x4 supercells. Ideally the smearing would be reduced further from the value used here, of 0.04, and the number of k-points increased within reason if the Gamma point only is determined to be insufficient.

This higher number of atoms per calculation would have the benefit of increasing the number of data points used to fit the potentials, as a consequence of increasing the number of force values. It would also allow the 1% doped value to be better represented by using 1, 2 or 3 atoms per 256 (0.4%, 0.8%, 1.2%) rather than 1 per 32 atoms (3%).

9.2.2 Alloy Bulk Properties

The original potentials used the bulk properties of the pure elements Fe, Pd and Ru. The calculations may then be performed using the QEEOS code to compute the bulk properties of Fe-Pd and Fe-Ru alloys. As the arrangement of atoms would not be unique it would not be as useful to use the figures output by QEEOS, but it would be more useful to use the files output by Quantum Espresso as additional configurations to add to the reference database.

9.2.3 Defects

In the derivation of the potentials in this work, more weighting was given to bulk and slab/surface configurations. There were also issues in SCF convergence for a number of the defect configurations. More effort could be put towards investigating how to ensure PWscf achieves convergence.

9.2.4 MEAM for FCC Iron

With a collinear spin-polarized DFT calculation, the relaxed structure was slightly longer in the y direction. Investigating the use of an angularly dependent potential might be worthwhile and the MEAM type potential with an angularly dependent electron density would be a starting point (section 5.3.3).

9.2.5 Improved Potential: Fe-Ni-Cr-Pd

Given more time and a larger computer, more complicated DFT calculations could be performed that better represent the material, austenitic stainless steel. As already mentioned, there are drawbacks (in terms of memory and computational time) and benefits in using a supercell with more atoms. For example a 1% PGM doped steel could be represented as a 4x4x4 supercell containing 256 atoms. For 304SS approximately 50 Cr, 25 Ni, 178 Fe and 3 Pd or 3 Ru atoms would be needed (and small variations on these figures).

9.2.6 PKA Cascade Trials in LAMMPS or DL_POLY

The first set of trials using the potential should be PKAs within a suitably sized simulation box, perhaps tens of thousands to hundreds of thousands of atoms. Individual cascades may be simulated, examined and compared to previous simulations as discussed earlier in this work. These would be the first steps towards testing how well the potential represents the material as well as the damage cascades by virtue of the ZBL function that forms a part of the pair functions.

9.2.7 Radiation Induced Segregation Trials

The ultimate goal for this part of the work is to determine whether or not PGMs are depleted at the grain boundary removing the protection to corrosion that they provide. There is a possibility of using the potentials with molecular dynamics, but the time period to cover the number of displacements required may be too large for classical molecular dynamics. An alternative use for the potential would be to explore atomic kinetic Monte Carlo (akMC) and this could allow simulation over a much longer period of time in order to reach damage doses up to 100 DPA.

Acronyms

2BEAM two band embedded-atom method. 68, 71, 78, 134, 135, 168, 175, 425, 499

AGR Advanced Gas Reactor. 6, 34

akMC atomic kinetic Monte Carlo. i, 75, 134, 169, 178

API Application Programming Interface. 78

ASME American Society of Engineers. 25

AVF Azimuthally Varying Field. 45

BCC body centered cubic. i, 12, 24–26, 32, 59, 60, 68, 69, 72, 85, 100, 134, 135, 142, 143, 147, 148, 157, 165, 166, 173, 174, 177, 489, 497, 498

BFGS Broyden Fletcher Goldfarb Shanno. 54, 57, 158, 160, 164, 416, 419, 420

BLYP Becke-Lee-Yang-Parr. 137

BWR Boiling Water Reactor. 9, 27, 34

BZ Brillouin zone. 85, 95, 96

CCGT Combined Cycle Gas Turbine. 3, 4, 7, 8

CERN Conseil Européen pour la Recherche Nucléaire. 40

CG Conjugate Gradient. 158, 160, 419

CPH face centered tetragonal. 100

CPU central processing unit. 79

DFT Density Functional Theory. i, 13, 54, 56, 60, 75, 77, 79, 80, 91, 94–97, 100, 134, 135, 137, 139, 141–145, 147, 148, 151–153, 155, 156, 158, 165, 167, 173–176, 178, 358, 361, 382, 416, 422, 483, 492

DNA Deoxyribonucleic Acid. 15

DPA displacements per atom. i, 12, 27, 29, 30, 103, 115, 126, 169, 170, 178, 333

EAM embedded-atom method. i, 12–14, 54, 67, 68, 70–72, 74, 79, 134, 135, 161, 164, 174, 175, 425

EAMPA Embedded Atom Method Potential Analyser. 134

EBR Experimental Breeder Reactor. 29

ECIS Equations Couplées Itérations Séquentielle. 46

ECP electrochemical corrosion potential. 34, 35

ELSY European Lead Fast Reactor. 7

ENDF Evaluated Nuclear Data File. 38, 48, 50

EPR European Pressurised Reactor. 4, 5

EXFOR EXchange FORmat. 45, 120, 121, 128–130

FATI Fusion Activation Transport Interface. 103, 133

FCC face centered cubic. i, 24–26, 32, 54, 59, 60, 68, 85, 100, 134, 135, 143–145, 147, 149, 150, 152, 154, 157, 158, 165, 166, 169, 173, 174, 177, 359, 481, 483–490, 492–496, 498

FCT face centered tetragonal. 147, 154, 174

FISPACT FISPACT. 103, 133

FS Finnis-Sinclair. 13, 14, 22, 54, 175

Gen II Generation II. 27, 29

Gen III+ Generation III+. i, 1, 10

Gen IV Generation IV. i, 1, 6, 9, 10, 27, 35, 42, 126, 170

GFR Gas-Cooled Fast Reactor. 8, 35

GGA generalized gradient approximation. 92–96, 137, 138, 141, 147, 361

GPU graphical processing unit. 75

GULP General Utility Lattice Program. 78

HCP hexagonal close packed. i, 134, 150, 177

HFIR High Flux Isotope Reactor. 40, 42

HK Hohenberg-Kohn. 86, 87, 95

HPGe High Purity Germanium. 127

IAEA International Atomic Energy Agency. 45

IASCC irradiation assisted stress corrosion cracking. 32

IBZ irreducible Brillouin zone. 85, 95

IGSCC inter granular stress corrosion cracking. i, 1, 24, 26, 31–33, 35, 169

IKE inverse Kirkendall effect. 29

INS Institute for Nuclear Study. 45

ITER International Thermonuclear Experimental Reactor. 10

JEFF Joint Evaluated Fission and Fusion File. 43, 50

KE Kirkendall effect. 29

KS Kohn-Sham. 91

LAMMPS Large-scale Atomic/Molecular Massively Parallel Simulator. 75–77, 161, 165, 169, 174, 176, 424, 481, 490

LDA local density approximation. 91–94, 137, 138, 141, 147, 361

LFR Lead-Cooled Fast Reactor. 6–8

LHC Large Hadron Collider. 40

LINAC linear accelerator. 38, 41

LM-BFGS Limited-memory Broyden Fletcher Goldfarb Shanno. 419

LMA Levenberg-Marquardt Algorithm. 78, 416, 420

LSDA local spin density approximation. 92–96, 141, 361

LWR Light Water Reactor. 34

MCNP Monte Carlo N-Particle. 16, 103, 133

MD molecular dynamics. i, 23, 54, 63, 65, 72, 75, 77, 134, 135, 169, 424

MEAM modified embedded-atom method. 71, 161, 174, 178

MOX mixed oxide. 76

MPI Message Parsing Interface. 75

MSKP Mehl, Singh, Klein, Papaconstantopoulos. 375, 485, 487, 489, 494, 496, 498

MSR Molten Salt Reactor. 9

NBS National Bureau of Standards. 40

NBSR National Bureau of Standards Reactor. 41

NEA Nuclear Energy Agency. 50

NG Newton-Gauss. 419, 420

NIST National Institute of Standards and Technology. 172

OMP optical model potential. 38

OpenMP Open Multi-Processing. 75

ORNL Oak Ridge National Laboratory. 40, 41

PADF Proton Activation Data File. 47

PBC periodic boundary conditions. 174

PBE Perdew-Burke-Ernzerhof. 93, 94, 138, 361–363

PBESOL Perdew-Burke-Ernzerhof functional revised for solids. 94

PBR Pebble-Bed Reactor. 8

- PEPT** Positron Emission Particle Tracking. 43
- PGM** platinum group metal. i, 1, 23, 35, 37, 101, 134, 135, 150, 154, 165, 169, 173, 174, 178
- PKA** primary knock-on atom. 11, 12, 22, 23, 26, 27, 72, 75–77, 178
- PWR** Pressurised Water Reactor. 9, 27, 34, 35
- PWscf** Plane-Wave Self-Consistent Field. 134, 137, 140, 144
- PZ** Perdew-Zunger. 94, 138
- QP** Quadratic Programming. 419
- RAM** Random Access Memory. 143
- RBE** Relative Biological Effectiveness. 20
- RFKJ** Ravindran, Fast, Korzhavyi, Johansson. 377
- RIS** radiation induced segregation. 23, 29–31, 34
- RSS** residual squared sum. 148, 149, 158, 197–202
- SA** Simulated Annealing. 419
- SCC** stress corrosion cracking. 29, 34, 37
- SCF** self consistent field. 89, 97, 140, 143, 145, 154, 156, 178, 404–407
- SCWR** Supercritical-Water-Cooled Reactor. 7–9
- SFR** Sodium-Cooled Fast Reactor. 7–9, 35
- SLAC** Stanford Linear Accelerator Center. 38
- SNS** Spallation Neutron Source. 38, 41, 42
- SRIM** Stopping Range In Matter. 12, 13, 38, 43, 51–53, 65, 66, 102, 103, 113, 115, 126, 129, 130, 170–172, 176
- TEM** transmission electron microscope. 37
- TENDL** TALYS-based Evaluated Nuclear Data Library. 44, 48, 50, 110, 118, 120, 121, 128, 129
- TGSCC** trans granular stress corrosion cracking. 33
- TISE** Time Independent Schrödinger Equation. 80–83, 86, 87, 91, 94
- TRIM** TRansport In Matter. 38, 51, 52
- TRISO** Tristructural Isotropic. 9
- TRIUMF** TRI-University Meson Facility. 39
- URL** Uniform Resource Locator. 133, 137
- VASP** Vienna Ab initio Simulation Package. 137
- VHTR** Very-High-Temperature Reactor. 7, 8
- VPI** vacancies per ion. 126
- XC** exchange-correlation. 90–92, 94
- ZBL** Ziegler-Biersack-Littmark. i, 12, 65, 66, 72, 78, 134, 163, 178, 477

Glossary

304SS Austenitic stainless steel: 18-20%Cr, 8-11%Ni. 26, 29, 32, 35, 37, 178

316SS Austenitic stainless steel: 16.5-18.5%Cr, 10-13%Ni, 2.0-2.5%Mo. 26, 32, 35

Al Aluminium $Z = 13$, $A_r = 26.98$. 74, 134, 141, 154

allotrope an element that exists in multiple forms/structures - the common example of allotropes are graphite (hexagonal sheets) and diamond (regular tetrahedral structure) for carbon. 25

antiferromagnetic neighbouring electrons orientate in opposite directions, cancelling out overall magnetism.
71

barn Unit for cross sectional area ($1.0 \times 10^{28} m^2$). 19

Bloch theorem electron wave functions may be written as a sum of plane waves $\psi_{\vec{k},n} \vec{r} = \sum c_{\vec{k}+\vec{G},n} \exp(i(\vec{k} + \vec{G})\vec{r})$. 85

Bohr magneton natural unit for magnetic moment $5.788 \times 10^5 eV T^{-1}$. 147

Brillouin Zone the Wigner-Seitz cell in reciprocal space. 85

C Carbon $Z = 6$, $A_r = 12.00$. 24, 31

Cauchy pressure $C_{12} - C_{44}$. 71

Co Cobalt $Z = 27$, $A_r = 58.93$. 1, 100, 118, 125

conv_thr The PWscf energy convergence threshold for self consistency[**inputpw**], the default value is 1.0e-6.
144

Cr Chromium $Z = 24$, $A_r = 52.00$. 1, 2, 24, 33, 34, 70, 71, 100, 135, 141, 169

Cs Caesium $Z = 55$, $A_r = 132.91$. 68, 69

Cu Copper $Z = 29$, $A_r = 63.55$. 29

diagonalization The PWscf choice of diagonalization method[**inputpw**] - David (Davidson) is the default method, but other methods including CG are available. 144

enthalpy sum of internal energy plus pressure times volume $H = E_{internal} + PV$. 70

etot_conv_thr The PWscf energy convergence threshold in Ry for ionic minimization, i.e. when running a relax or vc-relax problem[**inputpw**] - the default value is 1.0e-4. 144

eutectic A eutectic contains two or more elements, and has a lower melting point than any of it's constituent elements.. 7

Fe Iron $Z = 26$, $A_r = 55.85$. 1, 24, 54, 70, 99, 100, 103, 126, 132, 134, 135, 141, 143, 148, 155, 156, 158, 163, 165, 166, 169, 173, 176

Fermi energy the energy of the highest occupied state. 86

fermion half-integer spin particle - includes electrons, protons, neutrons. 90

forc_conv_thr The PWscf force convergence threshold in Ry/Bohr for ionic minimization, i.e. when running a relax or vc-relax problem[**inputpw**] - the default value is 1.0e-3. 144

Gray Gray (Gy) how much energy is deposited into matter and 1Gy is equal to 1 Joule per kg. 20

Hamiltonian the total energy of a system - an operator in quantum mechanics, $\hat{H} = \hat{T} + \hat{V} = \hat{H} = -\frac{\hbar^2}{2m}\nabla^2 + V(\vec{r}, t)$. 80

homogeneous electron gas quantum mechanical model of electrons distributed uniformly through space (jellium). 91

invar effect invar is an Fe Ni (64/36) alloy that has a low thermal expansion coefficient for a range of temperatures below its curie temperature, and this low expansion is the invar effect. 94

Ir Iridium $Z = 77$, $A_r = 192.22$. 35

isostructural transformation a change from one structure to another. 69

jellium quantum mechanical model of electrons distributed uniformly through space (homogeneous electron gas). 88, 91

Kohn-Sham equations $\hat{v}_{KS}(\vec{r}_e, \vec{r}_n) = v_{n-e}(\vec{r}_e, \vec{r}_n) + \int d^3r' \frac{\rho(\vec{r}'_e)}{|\vec{r}'_e - \vec{r}_e|} + v_{xc}[\rho](\vec{r}_e)$. 88–90

matrix diagonalization Take matrix A and find a diagonal matrix D such that $S^{-1}AS = D$. 95

mean free path average (mean) distance a projectile travels before reacting with the target. 19

mixing_beta The PWscf mixing factor during the self consistency iterations[**inputpw**]. The default value is 0.7. 144

mixing_mode The PWscf mixing mode[**inputpw**]. The default is plain - charge density Broyden mixing - TF (Thomas-Fermi screening) is recommended for highly homogeneous systems and local-TF for highly inhomogeneous systems. 144

Mo Molybdenum $Z = 42$, $A_r = 95.95$. 2, 24, 29, 103, 130–132, 171

Neel temperature temperature at which an antiferromagnetic becomes a paramagnet. 71

Ni Nickel $Z = 28$, $A_r = 58.69$. 22, 24, 54, 74, 100, 135, 169

O Oxygen $Z = 8$, $A_r = 16.00$. 31

observationally stable theoretically unstable but yet to be measured as unstable by experiment. 128

orthorhombic $a, b, c, \alpha = \beta = \gamma = 90^\circ$. 55

P Phosphorus $Z = 15$, $A_r = 30.97$. 29

Pauli exclusion principle half-integer spin particles cannot have the same quantum numbers. 90

Pauli vector sum of the three Pauli matrices $\vec{\sigma} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} + \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix} + \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$. 101

Pd Palladium $Z = 46$, $A_r = 106.42$. 2, 35, 54, 134, 135, 141, 144, 149, 155, 156, 163, 165

pi mesons Pi mesons are unstable particles made of two quarks and virtual pi mesons carry the strong nuclear force. There are four combinations of quarks that make the three types of pi meson: Π^0 up quark and up antiquark or down quark and up antiquark, Π^+ up quark and down antiquark, Π^- down quark and up antiquark.. 17

plane wave a wave of parallel planes of constant frequency that are normal to the direction of propagation, for example $\exp(i\vec{k}\vec{r})$. 85

Pt Platinum $Z = 78$, $A_r = 195.08$. 35

Rad Rad (Rad) this is another measure for absorbed dose and 1 rad is equal to 100 ergs per gram (1 rad = 1.0cGy = 0.01Gy). 20

Rontgen Equivalent Man Rontgen Equivalent Man (REM) - equivalent/effective dose - 1 REM equivalent is equal to $1\text{Rad} \times W_r$ (1cGy $1\text{Rad} \times W_r$) and 1 REM effective is equal to $1\text{Rad} \times W_r \times W_t$ (1cGy $1\text{Rad} \times W_r \times W_t$). 21

Ru Ruthenium $Z = 44$, $A_r = 101.07$. 2, 35, 54, 134, 135, 141, 150, 155, 156, 165, 174

sensitization The precipitation of chromium rich carbides at the grain boundaries of stainless steel. 31

Sievert Sievert (Sv) - equivalent/effective dose - this is the absorbed dose with radiation weighting (equivalent) and tissue weighting (effective) factors applied $1\text{Sv}(\text{equivalent}) = 1\text{J/kg} \times W_r$ and $1\text{Sv}(\text{effective}) = 1\text{J/kg} \times W_r \times W_t$. 21

supercritical Above 374°C (647K) and 22.1MPa water is no longer a distinct liquid or gas and it becomes supercritical. 9

TART A neutron transport code developed and maintained by Red Cullen at ORNL (Oak Ridge National Laboratory). 16

Tc Technetium $Z = 42$, $A_r = 96.91$. 130

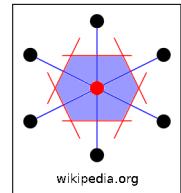
Th Thorium $Z = 90$, $A_r = 232.04$. 9

valence electron outer shell electron that can form bonds. 91

W Tungsten $Z = 74$, $A_r = 183.84$. 12

wave function contains all there is to know about a system and it's meaning is given by a probability $|\Psi|^2 = 1$. 80

Wigner-Seitz cell given a lattice point, the set of all points in space which are closer to that lattice point



that any other lattice point constitute the Wigner-Seitz cell[solidstatebasicswcs] wikipedia.org. 85

Young's modulus measure of resistance to elastic deformation $\delta = \frac{\text{stress}}{\text{strain}}$. 359

Zn Zinc $Z = 30$, $A_r = 65.38$. 29

Zr Zirconium $Z = 40$, $A_r = 91.22$. 5

Bibliography

- [1] H. Bateman. “The solution of a system of differential equations occurring in the theory of radioactive transformations”. In: *Proc. Camb. Phil. Soc.* 15 (1910), p. 423.
- [2] UK Government. *Historic Electricity Data*. URL: <https://www.gov.uk/government/statistical-data-sets/historical-electricity-data>.
- [3] Financial Times. *Cost of new Sizewell C Nuclear Plant*. 2020. URL: <https://www.ft.com/content/77c209f7-6d18-4609-ac3c-77d1b5b82b34>.
- [4] Energy for Generations. *ESB opens 820m Euro Carrington Power Station in Manchester*. 2020. URL: <https://www.esb.ie/tns/press-centre/2017/2017/03/13/esb-opens-820m-carrington-power-station-in-manchester>.
- [5] Our World In Data. *Safest and Cleanest Sources of Energy*. 2020. URL: <https://ourworldindata.org/uploads/2020/10/What-is-the-safest-form-of-energy-2048x1129.png>.
- [6] Yuchen Dou, Hong Luo, and Jing Zhang. “Elastic Properties of FeCr20Ni8Xn (X = Mo, Nb, Ta, Ti, V, W and Zr) Austenitic Stainless Steels: A First Principles Study”. In: *Metals* 9 (145 2019).
- [7] Philippe F Weck et al. *Mechanical properties of zirconium alloys and zirconium hydrides predicted from density functional perturbation theory*. 2020. URL: <https://www.osti.gov/pages/servlets/purl/1259850>.
- [8] GenIV International Forum. *Introduction to Generation IV Nuclear Energy Systems and the International Forum*. 2013. URL: <https://www.iea.org/topics/world-energy-outlook>.
- [9] Wikipedia. *Carnots theorem (thermodynamics)*. 2020. URL: <https://en.wikipedia.org/wiki/Carnot>.
- [10] S. E. Jensen and E. Nonbol. *Description of the Magnox Type of Gas Cooled Reactor (MAGNOX)*. 1998. URL: <https://inis.iaea.org/collection/NCLCollectionStore/\%5FPublic/30/052/30052480.pdf>.
- [11] Wikipedia. *Combined cycle power plant*. 2020. URL: <https://en.wikipedia.org/wiki/Combined\%5Fcycle\%5Fpower\%5Fplant>.
- [12] Los Alamos National Laboratory. *Criticality Safety*. URL: <https://t2.lanl.gov/nis/tour/sch007.html>.
- [13] World Nuclear. *Uranium Enrichment*. 2020. URL: <https://www.world-nuclear.org/information-library/nuclear-fuel-cycle/conversion-enrichment-and-fabrication/uranium-enrichment.aspx>.
- [14] James J Duderstadt and Louis J Hamilton. *Nuclear Reactor Analysis*. 1976.
- [15] Los Alamos National Laboratory. *Los Alamos National Laboratory Tour*. 2013. URL: <http://t2.lanl.gov/nis/tour/sch007.html>.
- [16] GenIV International Forum. *Very-High-Temperature Reactor (VHTR)*. 2020. URL: <https://www.gen-4.org/gif/jcms/c\%5F9362/vhtr>.

- [17] Sangeeta Deokattey et al. "Hydrogen production using high temperature reactors: an overview". In: *Advances in Energy Research* 1 (1 2013), pp. 013–033.
- [18] Leroy Cronin Greig Chisholm. *Hydrogen From Water Electrolysis*. 2016. URL: https://www.chem.gla.ac.uk/cronin/images/pubs/Chisholm-Chapter_16_2016.pdf.
- [19] James A. Mahaffey. *Atomic Accidents*.
- [20] S Heidenreich, M Müller, and P. U. Foscolo. *Advanced Biomass Gasification - New Concepts for Efficiency Increase and Product Flexibility*. 2016.
- [21] MIT Open Courseware. *Engineering of Nuclear Systems - Lecture 6A*. 2013. URL: <http://ocw.mit.edu/courses/nuclear-engineering/22-06-engineering-of-nuclear-systems-fall-2010/lectures-and-readings/MIT22\%5F06F10\%5Flec06a.pdf>.
- [22] MIT Open Courseware. *Engineering of Nuclear Systems - Lecture 6B*. 2013. URL: <http://ocw.mit.edu/courses/nuclear-engineering/22-06-engineering-of-nuclear-systems-fall-2010/lectures-and-readings/MIT22\%5F06F10\%5Flec06b.pdf>.
- [23] Guilherme Botelho Meireles de Souza et al. "Supercritical water technology: an emerging treatment process for contaminated wastewaters and sludge". In: *Reviews in Environmental Science and Bio/Techology* 21 (110 2022), pp. 75–104.
- [24] H. Khatabil. *Gen-4 Reactors*. 2013. URL: <http://www.gen-4.org/GIF/About/documents/25-Session2-3-Khartabil.pdf>.
- [25] V. Ignatiev and A. Surenkov. *Corrosion phenomena induced by molten salts in Generation IV nuclear reactors*. 2017, p. 166.
- [26] A. Santamarina et al. *The JEFF-3.1.1 Nuclear Data Library*. 2009.
- [27] M. R. Gilbert, J. Marian, and J-Ch. Sublet. "Energy spectra of primary knock-on atoms under neutron irradiation". In: *Journal of Nuclear Materials* 467 (2015), pp. 121–134.
- [28] Gary S. Was. *Challenges to the use of ion irradiation for emulating reactor irradiation*. URL: <https://link.springer.com/content/pdf/10.1557/jmr.2015.73.pdf>.
- [29] A. Souidi et al. "On the correlation between primary damage and long-term nanostructural evolution in iron under irradiation". In: *Journal of Nuclear Materials* 419 (2011), p. 122.
- [30] J. Fikar and R. Schaublin Ecole. "Molecular dynamics simulation of radiation damage in bcc tungsten". In: *Journal of Nuclear Materials* (101 2009), pp. 97–101.
- [31] Andrew F. Calder et al. "Computer simulation of cascade damage in a-iron with carbon in solution". In: *Journal of Nuclear Materials* (382 2008), pp. 91–95.
- [32] James F. Ziegler, M. D. Ziegler, and J. P. Biersack. "SRIM - The stopping and range of ions in matter". In: *Nuclear Instruments and Methods in Physics Research Section B* 268 (2010), pp. 1818–1823.
- [33] N J Carron. *An Introduction to the Passage of Energetic Particles through Matter*. 2007.
- [34] James Byrne. *Neutrons, Nuclei & Matter*. 2011, p. 161.
- [35] Douglas Fields. *Pair Production and Uncertainty*. 2020. URL: <https://physics.unm.edu/Courses/Fields/Phys2310/Lectures/lecture34.pdf>.
- [36] Douglas M. Gingrich. *Photon-Electron Scattering*. 2004. URL: <https://sites.ualberta.ca/~gingrich/courses/phys512/node102.html>.
- [37] Alex F Bielajew. *Photoelectric Effect*. 2001. URL: https://www.researchgate.net/figure/Photoelectric-effect_fig3_228538288.
- [38] G. F. Knoll. *Radiation Detection and Measurement Third Edition*. 1999, pp. 53–54.
- [39] NIST. *X-Ray Mass Attenuation Coefficients*. 2022. URL: <https://physics.nist.gov/PhysRefData/XrayMassCoef/tab3.html>.

- [40] J R Greening. *Fundamentals of Radiation Dosimetry*. 1985.
- [41] ICRP Publication 103.
- [42] *Radiation Factors*.
- [43] H. Sheng. *EAM Potentials*. 2012. URL: <https://sites.google.com/site/eampotentials>.
- [44] G. S. Was. *Fundamentals of Radiation Materials Science*. 2007.
- [45] University of Cambridge. *Mechanisms of Radiation Damage 1*. URL: https://www.doitpoms.ac.uk/tlplib/nuclear_materials/damage_mechanism.php.
- [46] University of Cambridge. *Mechanisms of Radiation Damage 2*. URL: https://www.doitpoms.ac.uk/tlplib/nuclear_materials/damage_mechanism2.php.
- [47] Can Erel et al. “Generation and interaction mechanisms of prismatic dislocation loops in FCC metals”. In: *Computational Materials Science* 140 (2017), p. 32.
- [48] R. E. Stoller. “Primary Damage Formation in Irradiated Materials”. In: *Journal of Metals* 48 (12 1996), pp. 23–27.
- [49] BSSA. *BSSA*. 2019. URL: https://bssa.org.uk/bssa_articles/the-schaeffler-and-delong-diagrams-for-predicting-ferrite-levels-in-austenitic-stainless-steel-welds.
- [50] Jian wei Su and Dong ping Huang. “Phase equilibria of the Fe–Cr–Ni ternary systems and interfacial reactions in Fe–Cr alloys with Ni substrate”. In: *Journal of Alloys and Compounds* 457 (2008), pp. 270–278.
- [51] Ben Palmer. *Photoelectric Effect*. 2020. URL: <https://github.com/BenPalmer1983/tikzcrystal>.
- [52] S. Choudhury a et al. “Ab-initio based modelling of diffusion in dilute bcc Fe–Ni and Fe–Cr alloys and implications for radiation induced segregation”. In: *Journal of Materials* 411 (2011), pp. 1–14.
- [53] G. S. Was. *Fundamentals of Radiation Materials Science*. 2007, p. 140.
- [54] J. L. Seran and M. Le Flem. *Irradiation-resistant Austenitic Steels as Core Materials for Generation IV Nuclear Reactors*. 2017, pp. 285–328.
- [55] T.R. Allen and J.T. Busby. “Radiation damage concerns for extended light water Reactor service”. In: *Journal of Materials* 61 (2009), pp. 29–34.
- [56] T. Wiss. *Material Properties/Oxide Fuels for Light Water Reactors and Fast Neutron Reactors*. Vol. 2. 2012.
- [57] Gary S. Was and Shigeharu Ukai. *Austenitic Stainless Steels*. 2019.
- [58] T.R. Allen et al. “Swelling and radiation-induced segregation in austentic alloys”. In: *Journal of Nuclear Materials* 342 (2005), pp. 90–100.
- [59] T. R. Allen, J. I. Cole, and E. A. Kenik. *Radiation-Induced Segregation and Void Swelling in 304 Stainless Steel*. 2000. URL: <https://www.osti.gov/servlets/purl/757550>.
- [60] M. Krcmar et al. “Diffusion rates of 3d transition metal solutes in nickel by first-principles calculations”. In: *Acta Materialia* 53 (2005), pp. 2369–2376.
- [61] IAEA Geir Meyer and Egil Stokke. *Description of Sizewell B Nuclear Power Plant*. 2019. URL: <https://inis.iaea.org/collection/NCLCollectionStore/\%5FPublic/29/010/29010110.pdf>.
- [62] G. S. Was. *Fundamentals of Radiation Materials Science*. 2007, p. 183.
- [63] B. Kombaiah et al. “Mechanisms of radiation-induced segregation around He bubbles in a Fe–Cr–Ni crystal”. In: *Journal of Nuclear Materials* 11 (2018).
- [64] W G Johnston, W G Morris, and A M Turkalo. “Proc. Int. Con. on Radiation Effects in Breeder Reactor Structural Materials”. In: *Metall. Soc. AIME* (1977), p. 421.

- [65] A D Marwick. "Segregation in irradiated alloys: The Inverse Kirkendall Effect and the effect of constitution on void swelling". In: *J. Phys. F: Met. Phys* 8 (1978), pp. 1849–1862.
- [66] P. R. Okamoto and L. E. Rehn. "RADIATION-INDUCED SEGREGATION IN BINARY AND TERNARY ALLOYS". In: *Journal of Nuclear Materials* 83 (1979), pp. 2–23.
- [67] Jeremy L. Gilber. *Metals - Basic Principles*.
- [68] R. Kirchheim et al. "Compositional Changes of Passive Films Due To Different Transport Rates and Preferential Dissolution". In: *Corrosion Science* 31 (1990), pp. 573–578.
- [69] R. Kirchheim et al. "The Passivity of Iron-Chromium Alloys". In: *Corrosion Science* 29 (7 1989), pp. 899–917.
- [70] P. J. Maziasz and M. Le Flem. *Properties of Austenitic Steels for Nuclear Reactor Applications*. Vol. 2. 2012, pp. 303–318.
- [71] Rolled Alloys. *What is alloy sensitization?* 2020. URL: <https://www.rolledalloys.com/dotAsset/50dc766c-08f5-43d9-bbda-5d9a9a2fa6d5.jpg>.
- [72] Speciality Steel Industry of North America. *Intergranular Corrosion*. 2020. URL: <https://www.ssina.com/education/corrosion/intergranular-corrosion/>.
- [73] S. Lyon. *Corrosion of Molybdenum and its Alloys*. 2010. URL: <https://www.researchgate.net/publication/287308921/Corrosion%5Fof%5FMolybdenum%5Fand%5Fits%5FAlloys>.
- [74] V. Maurice et al. "Effects of molybdenum on the composition and nanoscale morphology of passivated austenitic stainless steel surfaces". In: *Faraday Discussions* 180 (2015), pp. 151–170.
- [75] G. S. Was. *Fundamentals of Radiation Materials Science*. 2007, p. 765.
- [76] Edward Kenik, R H Jones, and G E C Bell. "Irradiation-assisted stress corrosion cracking". In: *Journal of Nuclear Materials* (212-215 1994), pp. 52–59.
- [77] Sophie Le Caër. *Water Radiolysis: Influence of Oxide Surfaces on H₂ Production under Ionizing Radiation*. 2011. URL: <https://www.mdpi.com/2073-4441/3/1/235>.
- [78] R.W. Staehle. *Historical views on stress corrosion cracking of nickel-based alloys: the Coriou effect*. 2016.
- [79] R.W. Staehle. *Historical views on stress corrosion cracking of nickel-based alloys: the Coriou effect*. 2016, p. 12.
- [80] R.W. Staehle. *Historical views on stress corrosion cracking of nickel-based alloys: the Coriou effect*. 2016, p. 15.
- [81] United States Nuclear Regulatory Commission. *AP1000 Design Control Document*. 2020. URL: <https://www.nrc.gov/docs/ML1117/ML11171A447.pdf>.
- [82] P. N. Standring. *The Long Term Storage of Advanced Gas-Cooled Reactor Fuel*. URL: <https://inis.iaea.org/collection/NCLCollectionStore/%5FPublic/30/040/30040095.pdf>.
- [83] Guy O.H. Whillock et al. "Investigation of thermally sensitised stainless steels as analogues for spent AGR fuel cladding to test a corrosion inhibitor for intergranular stress corrosion cracking". In: *Journal of Nuclear Materials journal* (498 2018), pp. 187–198.
- [84] IAEA. *Status report 81 - Advanced Passive PWR (AP 1000)*. 2020. URL: <https://aris.iaea.org/PDF/AP1000.pdf>.
- [85] Framatome ANP and Inc. *EPR Design Description*. 2020. URL: <https://www.nrc.gov/docs/ML0522/ML052280170.pdf>.
- [86] F. Dalle et al. *Conventional austenitic steels as out-of-core materials for Generation IV nuclear reactors*.
- [87] Pascal V. Grundler and Stefan Ritter. "Nobel Metal Chemical Addition for Mitigation of Stress Corrosion Cracking: Theoretical Insights and Applications". In: *PPChem* (2014), pp. 76–93.

- [88] J. H. Potgieter and A. Van Benekom. “The effect of varying ruthenium content on the corrosion behavior of two cathodically modified superferritic stainless steels”. In: *Canadian Metallurgical Quarterly* 34 (2 1994), pp. 143–146.
- [89] D. W. Saxy et al. “Understanding Stress Corrosion Resistant Stainless Steels at the Atomic Scale”. In: (2010).
- [90] IAEA Department of Nuclear Sciences and Applications. *Directory of Cyclotrons used for Radionuclide Production in Member States*. 2006. URL: <http://www-naweb.iaea.org/napc/iachem/cyclotrons/PDF/DCRP.pdf>.
- [91] Diamond. *About Synchrotrons*. 2020. URL: <https://www.diamond.ac.uk/Home/About/FAQs/About-Synchrotrons.html>.
- [92] World Nuclear. *World Nuclear Research Reactors*. 2018. URL: <http://www.world-nuclear.org/information-library/non-power-nuclear-applications/radioisotopes-research/research-reactors.aspx>.
- [93] IAEA. *Nepture Reactor*. 2018. URL: <https://nucleus.iaea.org/RRDB/RR/ReactorSearch.aspx>.
- [94] ORNL. *High Flux Isotope Reactor User Guide*. 2018. URL: <https://neutrons.ornl.gov/sites/default/files/HighFluxIsotopeReactorUserGuide2.0.pdf>.
- [95] ORNL. *High Flux Isotope Reactor Parameters*. 2018. URL: <https://neutrons.ornl.gov/hfir/parameters>.
- [96] ORNL. *High Flux Isotope Reactor (HFIR) USER GUIDE*. 2015. URL: <https://neutrons.ornl.gov/sites/default/files/High%20Flux%20Isotope%20Reactor%20User%20Guide%202.0.pdf>.
- [97] John J. Rush and Ronald L. Cappelletti. *The NIST Center for Neutron Research: Over 40 Years Serving NIST/NBS and the Nation*. 2015. URL: <https://www.ncnr.nist.gov/NCNRHistory%5FRush%5FCappelletti.pdf>.
- [98] ISIS. *How ISIS works - In Depth*. 2020. URL: <https://www.isis.stfc.ac.uk/Pages/How-ISIS-works--in-depth.aspx>.
- [99] ORNL E. B. Iverson. *Neutron Moderation*. 2016. URL: <https://conference.sns.gov/event/56/attachments/64/99/Lecture%5F2a%5F-%5FNeutron%5FModeration%5F-%5FErik%5FIverson.pdf>.
- [100] M. N. H. Comsan. *Spallation Neutron Sources for Science and Technology*. 2011. URL: <https://inis.iaea.org/collection/NCLCollectionStore/%5FPublic/43/099/43099436.pdf>.
- [101] R. B. Leachman. *Peaceful Uses of Atomic Energy*. Vol. 2. 1956, p. 193.
- [102] D. Parker and C. Wheldon. “The Birmingham MC40 Cyclotron Facility”. In: *Nuclear Physics News* 28 (4 2018), pp. 15–20.
- [103] Ben Palmer, Brian Connolly, and Mark Read. “Activity computer program for calculating ion irradiation activation”. In: *Computer Physics Communications* 216 (2017), pp. 138–144.
- [104] A.J. Koning and D. Rochman. “TENDL: Comprehensive Nuclear Data Library with Covariances”. In: (2010).
- [105] Hyperphysics. *Calculation of Coulomb Barrier*. 2022. URL: <http://hyperphysics.phy-astr.gsu.edu/hbase/NucEne/coubar.html>.
- [106] Hyperphysics. *Nuclear Size and Density*. 2022. URL: <http://hyperphysics.phy-astr.gsu.edu/hbase/Nuclear/nucuni.html#c4>.
- [107] V.V. Zerkin and B. Pritychenko. “The experimental nuclear reaction data (EXFOR): Extended computer database and Web retrieval system”. In: *Nuclear Inst. and Methods in Physics Research, A* 888 (2018), pp. 31–43.

- [108] Shigeo Tanaka and Michiaki Furukawa. “Excitation Functions for (p,n) Reactions with Titanium, Vanadium, Chromium, Iron and Nickel up to Ep”. In: *Journal of the Physical Society of Japan* 14 (10 1959), pp. 1269–1275.
- [109] M. S. Livingston and H. A. Bethe. “Nuclear Physics C. Nuclear Dynamics, Experimental”. In: *Physical Review* 9 (3 1937), pp. 245–398.
- [110] J. Wing and J. R. Huizenga. “(p,n) Cross Sections of V51, Cr52, Cu63, Cu65, Ag107, Ag109, Cd111, Cd112 and La139 from 5 to 10.5MeV”. In: *Physical Review* 128 (1 1962), pp. 280–290.
- [111] E. Gadioli et al. “Excitation Functions of V51, Fe56, 65Cu (p,n) Reactions Between 10 and 45 MeV”. In: *Il Nuovo Cimento* 22 (4 1974), pp. 547–558.
- [112] P E Hodgson. “The Nuclear Optical Model”. In: *Rep. Prog. Phys.* 34 (1971), pp. 765–819.
- [113] H Feshbach, C E Porter, and V F Weisskopf. “Model for Nuclear Reactions with Neutrons”. In: *Physical Review* 96 (1954), pp. 448–464.
- [114] A Koning, S Hilaire, and S Goriely. *TALYS 1.95 A Nuclear Reaction Program*. 2019.
- [115] NEA. *Janis Book*. 2020. URL: <https://tendl.web.psi.ch/tendl\%5F2019/other/book-protons.pdf>.
- [116] NEA. *Exfor reaction cross section website*. 2020. URL: <https://www-nds.iaea.org/exfor/>.
- [117] nuclear power.com. *Bateman Equation*. 2020. URL: <https://nuclear-power.com/wp-content/uploads/2018/10/Bateman-Equations-definition.png>.
- [118] J. F. Ziegler, J. P. Biersack, and M. D. Ziegler. *The Stopping and Range of Ions in Matter*. 2015.
- [119] Roger Rousseau, Vassiliki-Alexandra Glezakou, and Annabella Selloni. “Theoretical insights into the surface physics and chemistry of redox-active oxides”. In: *Nature Reviews Materials* (5 2020), pp. 460–475. URL: <https://www.nature.com/articles/s41578-020-0198-9?proof>.
- [120] F. D. Murnaghan. “The Compressibility of Media Under Extreme Pressures”. In: (1944).
- [121] A. Mahmoud ad A. Erba. *Crystal: Equation of State*. 2019. URL: <http://tutorials.crystalsolutions.eu/tutorial.html?td>.
- [122] F. Birch. “Finite elastic strain of cubic crystals”. In: *Physical Review* 71 (11 1947), pp. 809–824.
- [123] gilgamesh. *Fitting Birch-Murnaghan*. 2013. URL: <http://gilgamesh.cheme.cmu.edu/doc/software/jacapo/appendices/appendix-eos.html>.
- [124] M. J. Mehl, D. J. Singh, and D. A. Papaconstantopoulos. “Properties of ordered intermetallic alloys: first-principles and approximate methods”. In: *Materials Science and Engineering* (1993), pp. 49–57.
- [125] Pascal Vinet et al. “Universal features of the equation of state of solids”. In: *Journal of Physics Condensed Matter* 1 (1989), pp. 1941–1963.
- [126] G. Bonny, A. Bakaev, and D. Terentyev. “Interatomic potential to study plastic deformation in tungsten-rhenium alloys”. In: *Journal of Applied Physics* 121 (165107 2017), pp. 1–13.
- [127] B Jelinek et al. “Modified embedded atom method potential for Al, Si, Mg, Cu, and Fe alloys”. In: *Physical Review B* 84 (24 2012), pp. 1–18.
- [128] G. Bonny, R. C. Pasianot, and L. Malerba. “Interatomic potentials for alloys: Fitting concentration dependent properties”. In: *Philosophical Magazine* 89 (8 2009), pp. 711–725.
- [129] Michael J. Mehl. “Pressure dependence of the elastic moduli in aluminium-rich Al-Li compounds”. In: *Physical Review B* 47 (5 1993), pp. 2493–2500.
- [130] P. Ravindran et al. “Density functional theory for calculation of elastic properties of orthorhombic crystals applications to TiSi2”. In: *Journal of Applied Physics* 84 (9 1998), pp. 4891–4904.
- [131] M. J. Mehl, B. M. Klein, and D. A. Papaconstantopoulos. *First Principles Calculations of Elastic Properties of Metals*. 1994.

- [132] J. E. Jones. "On the determination of molecular fields.—I. From the variation of the viscosity of a gas with temperature". In: *Proceedings of the Royal Society A* 106 (738 1924).
- [133] Philip M. Morse. "Diatom Molecules According to the Wave Mechanics. II. Vibrational Levels". In: *Phys. Rev.* 34 (1929), p. 57.
- [134] Christopher M. Mauney and Davide Lazzati. "The formation of astrophysical Mg-rich silicate dust". In: *Molecular Astrophysics* 12 (March 2018), pp. 1–9.
- [135] M. W. Finnis and J. E. Sinclair. "A simple empirical N-body potential for transition metals". In: *Philosophical Magazine A* 50 (1984).
- [136] Murray S. Daw and M. I. Baskes. "Embedded-atom method: Derivation and application to impurities, surfaces and other defects in metals". In: *Physical Review B* 29 (12 1983).
- [137] Graeme J. Ackland. "Two-band second moment model for transition metals and alloys". In: *Condensed Materials* (2005).
- [138] P. Olsson et al. "Two-band modelling of alpha prime phase formation in Fe-Cr". In: *Physical Review B* 72 (2005).
- [139] G. Bonny et al. "Iron chromium potential to model high-chromium ferritic alloys". In: *Philosophical Magazine* 91 (April 2011), pp. 1724–1746.
- [140] Byeong-Joo Lee, Jae-Hyeok Shim, and M. I. Baskes. "Semiempirical atomic potentials for the fcc metals Cu, Ag, Au, Ni, Pd, Pt, Al, and Pb based on first and second nearest-neighbor modified embedded atom method". In: *Physical Review B* 68 (2003).
- [141] J. Fikar and R. Schäublin. "Molecular dynamics simulation of radiation damage in bcc tungsten". In: *Journal of Nuclear Materials* 386-388 (C 2009), pp. 97–101.
- [142] Y. Mishin. "Atomistic modeling of the gamma and gamma prime phases of the Ni-Al system". In: *Acta Materialia* 52 (2004), pp. 1451–1467.
- [143] M. I. Mendelev et al. "Development of new interatomic potentials appropriate for crystalline and liquid iron". In: *Philosophical Magazine* 83 (2003), pp. 3977–3994.
- [144] Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. 2006, p. 449.
- [145] F. Ercolessi and J. B. Adams. "Interatomic Potentials from First-Principles Calculations: the Force-Matching Method". In: *Europhys. Lett.* 26 (8 1994), pp. 583–588.
- [146] G. Bonny and L. Malerba. "Overview of Interatomic Potentials". In: *SCK CEB Open Report* (2005).
- [147] Hector Balboa et al. "Damage characterization of (U,Pu)O₂ under irradiation by molecular dynamics simulations". In: *Journal of Nuclear Materials* 512 (2018), pp. 440–449.
- [148] K. A. Nekrasov and. "Sputtering of material from the surface of PuO₂ crystals by collision cascades impact. A molecular dynamics study". In: *Journal of Nuclear Materials* 512 (2018), pp. 440–449.
- [149] "Defect production due to displacement cascades in metals as revealed by computer simulation". In: *Journal of Nuclear Materials* 251 (1997), pp. 1–12.
- [150] K. Trachenko et al. "Modeling high-energy radiation damage in nuclear and fusion applications". In: *Nuclear Instruments and Methods in Physics Research B* 277 (2012), pp. 6–13.
- [151] Derek J Hepburn and Graeme J Ackland. "Metallic-covalent interatomic potential for carbon in iron". In: *Physical Review B* 78 (165115 2008), pp. 1–6.
- [152] P. Brommer and F. Gähler. "Potfit: effective potentials from ab initio data". In: *Modelling Simul. Mater. Sci. Eng.* 15 (2007), pp. 295–304.
- [153] H. W. Sheng et al. "Highly Optimized Embedded-Atom-Method Potentials for Fourteen FCC Metals". In: *Europhys. Lett.* 26 (8 1994), pp. 583–588.

- [154] Julian Gale. "GULP: A computer program for the symmetry-adapted simulation of solids". In: *J. Chem. Soc., Faraday Trans.* 93 (4 1997), pp. 629–637.
- [155] D. Connetable and O. Thomas. "First-principles study of the structural, electronic, vibrational and elastic properties of orthorhombic NiSi". In: *Physical Review B* 79 (2009).
- [156] Materials. *Microstructure and Mechanical Properties of*. 2019. URL: <https://www.mdpi.com/1996-1944/12/2/290/pdf#:~:text>.
- [157] Steven H. Simon. *Solid State Basics*. 2013.
- [158] MIT. *Physics for Solid State Applications*. 2004. URL: <http://web.mit.edu/6.730/www/ST04/Lectures/Lecture7.pdf>.
- [159] Harry Jones. *The Theory of Brillouin Zones and Electronic States in Crystals*. 1960, p. 48.
- [160] P. Hohenberg and W. Kohn. "Inhomogeneous Electron Gas". In: *Physical Review* 136 (1964).
- [161] Robert van Leeuwen. *Introduction to density-functional theory*. 2020. URL: <http://users.jyu.fi/~roleeuwe/DFTLectureNotes.pdf>.
- [162] W. Kohn and L. J. Sham. "Self-Consistent Equations Including Exchange and Correlation Effects". In: *Physical Review* 140 (1965).
- [163] SISSA. *ABC of DFT*. 2020. URL: <https://people.sissa.it/~degironc/ES/lectures/ABCoDFT.pdf>.
- [164] J. F. Annett. "Efficiency of algorithms for Kohn-Sham density functional theory". In: *Computational Materials Science* 4 (1995), pp. 23–42.
- [165] G. Kresse and J. Furthmüller. "Efficient iterative schemes for ab initio total-energy calculations using a plane-wave basis set". In: *Physical Review B* 54 (16 1996).
- [166] D. Raczkowski, A. Canning, and L. W. Wang. "Thomas Fermi charge mixing for obtaining self consistency in DF calculations". In: *Physical Review B* 64 (2001).
- [167] D. D. Johnson. "Modified Broydens method for accelerating convergence in self-consistent calculations". In: *Physical Review B* 38 (18 1988).
- [168] P. Ziesche, S. Jurth, and J. P. Perdew. "Density functionals from LDA to GGA". In: *Computational Materials Science* 11 11 (1998), pp. 122–127.
- [169] M. C. Payne et al. "Iterative minimization techniques for ab initio total-energy calculations: molecular dynamics and conjugate gradients". In: *REVIEWS OF MODERN PHYSICS* 64 (1992).
- [170] Annabella Selloni. *The plane wave-pseudopotential method (basic aspects)*. 2009. URL: <https://th.fhi-berlin.mpg.de/th/Meetings/DFT-workshop-Berlin2009/Talks/OnlinePublication/0624-2\%5F20090623-1\%5FSelloni\%5FFINAL\%5F2perA4\%5F-\%5FSelloni-Berlin2.pdf>.
- [171] V.P. Gupta. *PRINCIPLES AND APPLICATIONS OF QUANTUM CHEMISTRY - Density Functional Theory (DFT) and Time Dependent DFT (TDDFT)*. 2016, p. 169.
- [172] John P Perdew, Kieron Burke, and Matthias Ernzerhof. "Generalized Gradient Approximation Made Simple". In: *Physical Review Letters* 77 (18 1996), pp. 3865–3868.
- [173] John P Perdew. "Generalized Gradient Approximations for exchange and correlation: A look backward and forward". In: *Physica B* 172 (1991), pp. 1–6.
- [174] M R Benam, N Abdoshahi, and M Majidiyan Sarmazdeh. "Ab initio study of the effect of pressure on the structural and electronic properties of cubic LaAlO₃ by density function theory using GGA, LDA and PBESOL exchange correlation potentials". In: *Physica B* 446 (2014), pp. 32–38.
- [175] periodictable.com. *Sodium: periodictable.com*. 2020. URL: <https://periodictable.com/Elements/011/data.html>.
- [176] periodictable.com. *Aluminium: periodictable.com*. 2020. URL: <https://periodictable.com/Elements/013/data.html>.

- [177] P Giannozzi et al. “Quantum Espresso”. In: *J Phys Condens Matter* 29 (2017).
- [178] University of Illinois. *Density functional theory calculation of elastic constants*. 2020. URL: <https://courses.physics.illinois.edu/mse404ela/sp2018/Project2-DFT.html>.
- [179] Alessandro Genova and Michele Pavanello. “Exploiting the Locality of Periodic Subsystem Density-Functional Theory: Efficient Sampling of the Brillouin Zone”. In: *J. Phys.: Condens. Matter* 27 (2015), pp. 495–501. URL: <https://iopscience.iop.org/article/10.1088/0953-8984/27/49/495501/ampdf>.
- [180] N. Marzari. *Ab-initio Molecular Dynamics for Metallic Systems*. 1996, p. 77.
- [181] Hyperphysics. *Electron band gap diagram for insulators, semiconductors and conductors*. 2020. URL: <http://hyperphysics.phy-astr.gsu.edu/hbase/Solids/band.html>.
- [182] University of Cambridge. *Dissemination of IT for the Promotion of Materials Science*. 2020. URL: <https://www.doitpoms.ac.uk/tlplib/semiconductors/fermi.php>.
- [183] M. Methfessel and A. T. Paxton. “High-precision sampling for Brillouin-zone integration in metals”. In: *Physical Review B* 40 (6 1989), pp. 3616–3621.
- [184] N. Marzari. *Ab-initio Molecular Dynamics for Metallic Systems*. 1996, p. 136.
- [185] Steven H. Simon. *Solid State Basics*. 2013, p. 213.
- [186] U Von Barth and L Hedin. “A local exchange-correlation potential for the spin polarized case”. In: *Journal of Physics C Solid State Physics* 5 (1972).
- [187] Edward A. Pluhar III and Carsten A. Ullrich. “Exchange-correlation magnetic fields in spin-density-functional theory”. In: *Physical Review B* 100 (12-15 September 2019 2019).
- [188] Gustav Bihlmayer. *Magnetism in Density Functional Theory*. 2020. URL: <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.563.8752&rep=rep1&type=pdf>.
- [189] R. A. Forrest and J. Kopecky. “The activation system EASY-2007”. In: *Journal of Nuclear Materials* 386 (2009).
- [190] L. W. G. Morgan et al. “The development of a fusion specific depletion interface code - FATI”. In: *Fusion Engineering and Design* 88 (2013), p. 2891.
- [191] J.-Ch. Sublet et al. “FISPACT-II: An Advanced Simulation System for Activation, Transmutation and Material Modelling”. In: *Nuclear Data Sheets* 139 (2017), p. 77.
- [192] Logan J. Harr. *Precise Calculation of Complex Radioactive Decay Chains*. 2007. URL: <https://scholar.afit.edu/etd/2924>.
- [193] *Bromwich Integral*.
- [194] H. Stehfest. “Algorithm 368 Numerical Inversion of Laplace Transform”. In: *Communications of the ACM* 13 (1970), pp. 47–49.
- [195] J. Ziegler. *The Stopping and Range of Ions in Matter*. 2019. URL: <http://www.srim.org/SRIM/SRIMINTRO.htm>.
- [196] A.J. Koning, D. Rochman, and J. Sublet. “TENDL: Comprehensive Nuclear Data Library with Covariances”. In: (2017).
- [197] A.J. Koning et al. “TENDL: Complete Nuclear Data Library for Innovative Nuclear Science and Technology”. In: *Nuclear Data Sheets* 155 (2019).
- [198] M.C.Lagunas-Solar and J.A.Jungerman. “Cyclotron production of carrier-free cobalt-55.a new positron-emitting label for bleomycin”. In: *Applied Radiation and Isotopes* 30 (1979), p. 25.
- [199] D. Rochman and A. J. Koning. “Modern Nuclear Data Evaluation With The TALYS Code System”. In: *Nuclear Data Sheets* 113 (2012).

- [200] J. Hewett. *Use of the University of Birmingham's Scanditronix MC40 Cyclotron for Materials Irradiation Damage Studies*. 2011.
- [201] E. Lamere et al. "Proton-induced reactions on molybdenum". In: *Physical Review, Part C, Nuclear Physics* 100 (2019), p. 034614.
- [202] D.S.Flynn, R.L.Hershberger, and F.Gabbard. "Sub-Coulomb proton absorption for isotopes of zirconium and molybdenum". In: *Physical Review, Part C, Nuclear Physics* 20 (5 1979), p. 1700.
- [203] J.J.Hogan. "Mo-96(P,XN)Reaction from 10 to 80 MeV". In: *Physical Review, Part C, Nuclear Physics* 6 (1972), p. 810.
- [204] V.N.Levkovski. "Mo-96(P,XN)Reaction from 10 to 80 MeV". In: *Soviet Atomic Energy* 69 (1990), p. 773.
- [205] Andrea Fortini, Mikhail I Mendelev, and Sergey Buldyrev et al. "Asperity contacts at the nanoscale: Comparison of Ru and Au". In: *J. Appl. Phys.* 104 (074320 2008), pp. 1–9.
- [206] Yuwen Zhang Seyed Moein Rassoulinejad-Mousavi. *Interatomic Potentials Transferability for Molecular Simulations: A Comparative Study for Platinum, Gold and Silver*. 2018. URL: <https://www.nature.com/articles/s41598-018-20375-4>.
- [207] webelements.com. *Webelements: Aluminium*. 2020. URL: <https://www.webelements.com/aluminium>.
- [208] webelements.com. *Webelements: Iron*. 2020. URL: <https://www.webelements.com/iron>.
- [209] webelements.com. *Webelements: Nickel*. 2020. URL: <https://www.webelements.com/nickel>.
- [210] webelements.com. *Webelements: Ruthenium*. 2020. URL: <https://www.webelements.com/ruthenium>.
- [211] webelements.com. *Webelements: Palladium*. 2020. URL: <https://www.webelements.com/palladium>.
- [212] N. Marzari et al. "Thermal Contraction and Disordering of the Al(110) Surface". In: *Condensed Matter* (110 2008), pp. 4–7.
- [213] A. L. Okana-Lomanga et al. "Adsorption of Br₂ molecule on the Fe W(110) substrate - Energetics, electronic and magnetic properties". In: *Computational Condensed Matter* 23 (2020).
- [214] A. E. Rezaee and M. Almasi-Kashi. "The influence of point defects on Na diffusion in black phosphorene - first principles study". In: *Journal Pre-proof* ().
- [215] N. Acharya and S. P. Sanyal. "Structural phase transition, electronic and superconducting properties of ScBi and YBi". In: *Solid State Communications* 266 (2017), pp. 39–45.
- [216] A. Choudhary, L. Malakkal, and R. K. Siripurapu. "First principles calculations of hydrogen storage on Cu and Pd-decorated graphene". In: *International Journal of Hydrogen Energy* 41 (2016), pp. 17652–17656.
- [217] O. E. Osafule, P. O. Adebambo, and G. A. Adebayo. "Elastic constants and observed ferromagnetism in inverse Heusler alloy Ti₂CoAs using kjpaw pseudopotentials - A first-principles approach". In: *Journal of Alloys and Compounds* 722 (2017), pp. 207–211.
- [218] Quantum Espresso. *QE Forum*. 2022. URL: <https://lists.quantum-espresso.org/>.
- [219] *Materials Square Single Atom Calculations*.
- [220] *Harwell VA08 Minimizer*. URL: <https://www.hsl.rl.ac.uk/archive/specs/va08.pdf>.
- [221] Christopher O'Brien. *Formatting EAM Potentials*. 2021. URL: <https://sites.google.com/a/ncsu.edu/cjobrien/tutorials-and-guides/eam>.
- [222] G.J. Ackland et al. "Computer simulation of point defect properties in dilute Fe-Cu alloy using a many-body interatomic potential". In: *Philosophical Magazine A* 75 (3 1997), pp. 713–732.
- [223] MIT. *Constitutive Equations*. 2020. URL: http://web.mit.edu/16.20/homepage/3_Constitutive_Constitutive_files/module_3_with_solutions.pdf.

- [224] B B Karki, G J Ackland, and J Crain. *Elastic instabilities in crystals from ab initio stress–strain relations*. 2013. URL: <http://www.homepages.ed.ac.uk/gja/karki1.pdf>.
- [225] G. Ghosh. “A first-principles study of cementite (Fe₃C) and its alloyed counterparts: Elastic constants, elastic anisotropies, and isotropic elastic moduli”. In: *AIP Advances* 5 (2015).
- [226] M. E. Fine, L. D. Brown, and H. L. Marcus. “Elastic Constants Versus Melting Temperature in Metals”. In: *Scripta Metallurgica* 18 (1984), pp. 951–956.
- [227] S. Kirkpatrick, C. D. Gelatt Jr, and M. P. Vecchi. “Optimization by Simulated Annealing”. In: *Science* 220 (4598 1983), pp. 671–680.
- [228] David J. Wales and Jonathan P. K. Doye. “Global Optimization by Basin-Hopping and the Lowest Energy Structures of Lennard-Jones Clusters Containing up to 110 Atoms”. In: *J. Phys. Chem. A* 1997 101 (1997), pp. 5111–5116.
- [229] *SciPy Basin Hopping Algorithm*.
- [230] Jonathan Richard Shewchuk. *An Introduction to the Conjugate Gradient Method Without the Agonizing Pain Edition 1.25*. 1994. URL: <https://www.cs.cmu.edu/~quake-papers/painless-conjugate-gradient.pdf>.
- [231] I.T.Todorov and W. Smith. *The DL_POLY_4 User Manual*. 2016.
- [232] *LAMMPS - Partitioning*. URL: https://docs.lammps.org/Developer_par_part.html.
- [233] Wikipedia. *Virial Stress*. 2016. URL: <https://en.wikipedia.org/wiki/Virial\%5Fstress>.

Chapter 10

Acknowledgements

Thank you to Brian, Mark and Alessandro for your supervision and support, to my wife Rachel, son Zakk, parents and my family for encouragement. The various forums in the scientific community, in particular useful discussions with Dr. Todorov of Daresbury and members of the Quantum Espresso forum, have been helpful throughout. I should also thank two teachers from my school days, Mr. George and Mr. Rockett, for their encouragement in the sciences (especially Physics).

The computations described in this work were performed using the University of Birmingham's BlueBEAR HPC service, which provides a High Performance Computing service to the University's research community. See <http://www.birmingham.ac.uk/bear> for more details.

Appendices

Appendix A

Useful Laplace Transforms

A.1 Table of Transforms

$f(t) = \mathcal{L}^{-1}\{F(s)\}$	$F(s) = \mathcal{L}\{f(t)\}$
$f(t)$	$F(s)$
$f'(t)$	$sF(s) - f(0)$
a	$\frac{1}{s+a}$
$\exp(-at)$	$\frac{1}{s+a}$
$a \exp(-bt)$	$\frac{a}{s+b}$
$\frac{1}{a} [1 - \exp(-at)]$	$\frac{1}{s(s+a)}$
$\frac{a}{b} [1 - \exp(-bt)]$	$\frac{a}{s(s+b)}$

Table A.1: Useful Laplace transforms

A.2 Decay with a Source Term

Set up the equation:

$$\frac{dN(t)}{dt} = \omega - \lambda N(t) \quad (\text{A.1})$$

Transform:

$$\begin{aligned} \mathcal{L} \left\{ \frac{dN(t)}{dt} \right\} &= sF(s) - N_0 \\ \mathcal{L} \{\omega - \lambda N(t)\} &= \frac{\omega}{s} - \lambda N(s) \end{aligned} \quad (\text{A.2})$$

Rearrange to make $N(s)$ the subject:

$$\begin{aligned}
 sN(s) - N_0 &= \frac{\omega}{s} - \lambda N(s) \\
 (s + \lambda)N(s) &= \frac{\omega}{s} + N_0 \\
 N(s) &= \frac{\omega}{s(s + \lambda)} + \frac{N_0}{(s + \lambda)}
 \end{aligned} \tag{A.3}$$

Inverse transform:

$$\begin{aligned}
 \mathcal{L}^{-1}\{N(s)\} &= N(t) \\
 \mathcal{L}^{-1}\left\{\frac{\omega}{s(s + \lambda)}\right\} &= \frac{\omega}{\lambda} (1 - \exp(-\lambda t)) \\
 \mathcal{L}^{-1}\left\{\frac{N_0}{s + \lambda}\right\} &= N_0 \exp(-\lambda t)
 \end{aligned} \tag{A.4}$$

Simplify:

$$\begin{aligned}
 N(t) &= \frac{\omega}{\lambda} (1 - \exp(-\lambda t)) + N_0 \exp(-\lambda t) \\
 N(t) &= \frac{\omega}{\lambda} + \left(N_0 - \frac{\omega}{\lambda}\right) \exp(-\lambda t)
 \end{aligned} \tag{A.5}$$

Appendix B

Activity Equation in Python: decay class

B.1 Full Source Code

The full source code is available to download from github.

https://github.com/BenPalmer1983/atomic_dictionaries

<https://github.com/BenPalmer1983/decay>

B.2 Highlighted Code

The code below is taken from the decay class and it is specifically responsible for calculating the amount of isotopes in the decay chain at some time t.

Listing B.1: Decay python3 class for calculating the amount of an isotope in a decay chain at time t

```
1 import os
2 import numpy
3 from pz import pz
4 from isotopes import isotopes
5 import matplotlib.pyplot as plt
6 import copy
7 import hashlib
8
9 class decay:
10
11     path_isotopes = "../data/isotopes.pz"
12     loaded = False
13
14     @staticmethod
15     def set(path_isotopes):
16         decay.path_isotopes = path_isotopes
17         isotopes.set(path_isotopes)
18         decay.load()
19
20     @staticmethod
21     def load():
22         if(decay.loaded == False):
23             decay.loaded = True
```

```

24
25
26     @staticmethod
27     def chain_isotopes(key, out=[]):
28         if(not isotopes.is_valid(key)):
29             return out
30         if(key not in out):
31             out.append(key)
32         else:
33             return out
34         if(isotopes.is_stable(key)):
35             return out
36         else:
37             dm = isotopes.get_decay_modes(key)
38             for k in dm.keys():
39                 decay.chain_isotopes(k, out)
40             return out
41
42
43     @staticmethod
44     def make_chain(key, l=0, out=[], bf=0.0):
45         if(not isotopes.is_valid(key)):
46             return out
47         if(l == 0):
48             decay.chains_store = []
49             out.append([l, key, bf])
50         if(isotopes.is_stable(key)):
51             if(len(out) > 1):
52                 for i in range(len(out)-1,1,-1):
53                     if(out[i-1][0]>=out[i][0]):
54                         out.pop(out[i-1][0])
55             return decay.chains_store.append(copy.deepcopy(out))
56         else:
57             l = l + 1
58             dms = isotopes.get_decay_modes(key)
59             for k in dms.keys():
60                 bf = dms[k]['branching_factor']
61                 decay.make_chain(k, l, out, bf)
62             return out
63
64     @staticmethod
65     def calculate(parent, time, i_data_in, log=None, custom_chain=None):
66
67         if(log != None):
68             log_dir = decay.get_file_dir(log)
69             print(log_dir)
70             decay.make_dir(log_dir)
71
72         decay.results = {
73             'tally': {},
74             'unique': None,
75             'chains': None,
76         }
77         decay.chains_store = None
78
79         if(custom_chain == None):
80             decay.chains_store = []
81             decay.make_chain(parent, 0, [])
82             decay.results['chains'] = []
83             cn = 0
84             for chain in decay.chains_store:

```

```

85     decay.results['chains'].append([])
86     for iso in chain:
87         k = iso[1]
88         bf = iso[2]
89         i_data = isotopes.get(k)
90         half_life = None
91         decay_constant = None
92         n0 = 0.0
93         w = 0.0
94         if(not i_data['stable']):
95             half_life = i_data['half_life']
96             decay_constant = i_data['decay_constant']
97             if(k in i_data_in.keys()):
98                 n0 = i_data_in[k]['n0']
99                 w = i_data_in[k]['w']
100            d = {
101                'isotope_key': k,
102                'bf': bf,
103                'w': w,
104                'n0': n0,
105                'nend': 0,
106                'half_life': half_life,
107                'decay_constant': decay_constant,
108            }
109            decay.results['chains'][cn].append(d)
110            cn = cn + 1
111        else:
112            decay.results['chains'] = custom_chain
113
114
115 # Find unique and make tally
116 decay.results['unique'] = []
117 for chain in decay.results['chains']:
118     for iso in chain:
119         k = iso['isotope_key']
120         if(k not in decay.results['tally'].keys()):
121             decay.results['unique'].append(k)
122             # Use provided data
123             if(iso['half_life'] == None):
124                 stable = True
125                 half_life = None
126                 decay_constant = 0.0
127             else:
128                 stable = False
129                 half_life = iso['half_life']
130                 decay_constant = numpy.log(2) / iso['half_life']
131                 w = iso['w']
132                 n0 = iso['n0']
133                 # Get proton/neutron etc from isotopes database
134                 i_data = isotopes.get(k)
135                 if(i_data is None):
136                     decay.results['tally'][k] = {
137                         'printable': 'Custom',
138                         'element': 'ZZ',
139                         'protons': 999,
140                         'nucleons': 999,
141                         'metastable': 9,
142                         'stable': stable,
143                         'half_life': half_life,
144                         'decay_constant': decay_constant,
145                         'w': w,

```

```

146             'n0': n0,
147             'nend': 0.0,
148         }
149     else:
150         decay.results['tally'][k] = {
151             'printable': decay.pad(isotopes.get_printable_name(k), 12),
152             'element': i_data['element'],
153             'protons': i_data['protons'],
154             'nucleons': i_data['nucleons'],
155             'metastable': i_data['metastable'],
156             'stable': stable,
157             'half_life': half_life,
158             'decay_constant': decay_constant,
159             'w': w,
160             'n0': n0,
161             'nend': 0.0,
162         }
163
164
165 decay.results['chains_individual'] = []
166 for cn in range(len(decay.results['chains'])):
167     for n in range(len(decay.results['chains'][cn])):
168         nc = []
169         if(decay.results['chains'][cn][n]['n0']>0.0 or decay.results['chains'][cn][n]['w']>0.0):
170             for j in range(n, len(decay.results['chains'][cn])):
171                 iso = copy.deepcopy(decay.results['chains'][cn][j])
172                 if(j>n):
173                     iso['n0'] = 0.0
174                     iso['w'] = 0.0
175                 nc.append(iso)
176         if(len(nc)>0):
177             decay.results['chains_individual'].append(nc)
178
179 for cn in range(len(decay.results['chains_individual'])):
180     chain = decay.results['chains_individual'][cn]
181     n0 = numpy.zeros((len(chain),),)
182     w = numpy.zeros((len(chain),),)
183     l = numpy.zeros((len(chain),),)
184     b = numpy.zeros((len(chain)-1,),)
185     for n in range(len(decay.results['chains_individual'][cn])):
186         k = decay.results['chains_individual'][cn][n]['isotope_key']
187         n0[n] = decay.results['chains_individual'][cn][n]['n0']
188         w[n] = decay.results['chains_individual'][cn][n]['w']
189         l[n] = decay.results['tally'][k]['decay_constant']
190         if(n>0):
191             b[n-1] = decay.results['chains_individual'][cn][n]['bf']
192         # Some observationally stable will still have a decay constant, so set to -1.0
193         if(isotopes.is_stable(k)):
194             l[n] = -1.0
195     nt = decay.calculate_activity(time, l, b, w, n0)
196     for n in range(len(decay.results['chains_individual'][cn])):
197         decay.results['chains_individual'][cn][n]['nend'] = nt[n]
198
199 set = []
200 for cn in range(len(decay.results['chains_individual'])):
201     ckey = ''
202     for n in range(len(decay.results['chains_individual'][cn])):
203         k = decay.results['chains_individual'][cn][n]['isotope_key']
204         ckey = ckey + str(decay.results['chains_individual'][cn][n]['isotope_key']) + "NO:" + str(decay.results['chains_individual'][cn][n]['n0']) + "W:" + str(decay.results['chains_individual'][cn][n]['w'])
205     ckeyh = hashlib.md5(ckey.encode())

```

```

206     ckeyh = ckeyh.hexdigest()
207     if(ckeyh not in set):
208         decay.results['tally'][k]['nend'] = decay.results['tally'][k]['nend'] + decay.results['chains_individual'
209             ][cn][n]['nend']
210         set.append(ckeyh)
211
212     # Log
213     if(log != None):
214         width = 140
215         fh = open(log, 'w')
216         fh.write("Unique Isotopes\n")
217         fh.write(decay.hr(width) + "\n")
218         for k in decay.results['unique']:
219             line = decay.results['tally'][k]['printable']
220             fh.write(line + "\n")
221             fh.write("\n")
222             fh.write("\n")
223             fh.write("Decay Chains\n")
224             fh.write(decay.hr(width) + "\n")
225             fh.write("\n")
226             fh.write("\n")
227             for cn in range(len(decay.results['chains'])):
228                 chain = decay.results['chains'][cn]
229
230                 fh.write(decay.pad(cn+1,6))
231                 for n in range(len(chain)):
232                     iso = chain[n]
233                     k = iso['isotope_key']
234                     if(n>0):
235                         bf = "{0:3e}".format(iso['bf'])
236                         fh.write(" --[" + str(bf) + "]--> ")
237                         fh.write(decay.pad(decay.results['tally'][k]['printable'], 12))
238                         fh.write("\n")
239                         fh.write(decay.pad("T1/2",6))
240                         for n in range(len(chain)):
241                             if(n>0):
242                                 fh.write(decay.pad("",17))
243                                 if(decay.results['chains'][cn][n]['half_life'] == None):
244                                     fh.write(decay.pad("[Stable]",12))
245                                 else:
246                                     fh.write(decay.pad("[" + str("{0:8e}".format(decay.results['chains'][cn][n]['half_life'])) + "]",12))
247                                     fh.write("\n")
248                                     fh.write(decay.pad("L",6))
249                                     for n in range(len(chain)):
250                                         if(n>0):
251                                             fh.write(decay.pad("",17))
252                                             if(decay.results['chains'][cn][n]['decay_constant'] == None):
253                                                 fh.write(decay.pad("[Stable]",12))
254                                             else:
255                                                 fh.write(decay.pad("[" + str("{0:8e}".format(decay.results['chains'][cn][n]['decay_constant'])) + "]",12))
256                                                 fh.write("\n")
257                                                 fh.write("\n")
258                                                 fh.write("\n")
259
260
261     fh.write("Amounts\n")
262     fh.write(decay.hr(width) + "\n")
263     fh.write("\n")

```

```

265     fh.write(decay.hr(width) + "\n")
266     line = decay.pad("Isotope", 12)
267     line = line + decay.pad("T(1/2)", 18)
268     line = line + decay.pad("Decay Constant", 18)
269     line = line + decay.pad("W", 18)
270     line = line + decay.pad("N(t=0)", 18)
271     line = line + decay.pad("N(t=" + str(time) + ")", 18)
272     line = line + decay.pad("A(t=0)", 18)
273     line = line + decay.pad("A(t=" + str(time) + ")", 18)
274     fh.write(line + "\n")
275     fh.write(decay.hr(width) + "\n")
276
277
278     for k in decay.results['tally'].keys():
279         line = decay.pad(decay.results['tally'][k]['printable'], 12)
280         if(decay.results['tally'][k]['half_life'] is None):
281             line = line + decay.pad("Stable", 18)
282             line = line + decay.pad("Stable", 18)
283         else:
284             line = line + decay.pad(str("{0:16e}").format(decay.results['tally'][k]['half_life'])).strip(), 18)
285             line = line + decay.pad(str("{0:16e}").format(decay.results['tally'][k]['decay_constant'])).strip(), 18)
286             line = line + decay.pad(str("{0:16e}").format(decay.results['tally'][k]['w'])).strip(), 18)
287             line = line + decay.pad(str("{0:16e}").format(decay.results['tally'][k]['n0'])).strip(), 18)
288             line = line + decay.pad(str("{0:16e}").format(decay.results['tally'][k]['nend'])).strip(), 18)
289         if(decay.results['tally'][k]['half_life'] is not None):
290             line = line + decay.pad(str("{0:16e}").format(decay.results['tally'][k]['decay_constant'] * decay.results
291                 ['tally'][k]['n0'])).strip(), 18)
292             line = line + decay.pad(str("{0:16e}").format(decay.results['tally'][k]['decay_constant'] * decay.results
293                 ['tally'][k]['nend'])).strip(), 18)
294         fh.write(line + "\n")
295     fh.write(decay.hr(width) + "\n")
296     fh.write("\n")
297     fh.write("\n")
298
299     return decay.results
300
301 ######
302 # DECAY EQUATIONS
303 #####
304
305 @staticmethod
306 def calculate_activity(t, lam, b, w, n0):
307     nt = numpy.zeros((len(n0),))
308     for m in range(0,len(n0)):
309         if(lam[m] <= 0):
310             nt[m] = decay.calc_stable(t, m, lam, b, w, n0)
311         else:
312             nt[m] = decay.calc_unstable(t, m, lam, b, w, n0)
313     return nt
314
315 @staticmethod
316 def calc_unstable(t, m, lam, b, w, n0):
317     y = 0.0
318     k = 0
319     while(k<=m):
320         y = y + decay.r(k, m, lam, b) * (decay.f_unstable(t, k, m, lam) * n0[k] + decay.g_unstable(t, k, m, lam) * w
321             [k])
322         k = k + 1
323     return y

```

```

323
324     @staticmethod
325     def f_unstable(t, k, m, lam):
326         s = 0.0
327         i = k
328         while(i<=m):
329             p = decay.lprod(lam, k, m, 3, i)
330             s = s + numpy.exp(-lam[i] * t) * (1.0 / p)
331             i = i + 1
332             s = s * (-1)**(m-k)
333         return s
334
335
336     @staticmethod
337     def g_unstable(t, k, m, lam):
338         # Term a
339         p = decay.lprod(lam, k, m, 1)
340         a = 1.0 / p
341         # Term b
342         s = 0.0
343         i = k
344         while(i<=m):
345             p = decay.lprod(lam, k, m, 3, i)
346             s = s + (1 / lam[i]) * numpy.exp(-lam[i] * t) * (1.0 / p)
347             i = i + 1
348             s = s * (-1)**(m-k+1)
349         return a + s
350
351     @staticmethod
352     def calc_stable(t, m, lam, b, w, n0):
353         y = n0[m] + w[m] * t
354         k = 0
355         while(k<=m-1):
356             y = y + decay.r(k, m, lam, b) * (decay.f_stable(t, k, m, lam) * n0[k] + decay.g_stable(t, k, m, lam) * w[k])
357             k = k + 1
358         return y
359
360     @staticmethod
361     def f_stable(t, k, m, lam):
362         mm = m - 1
363         p = decay.lprod(lam, k, mm, 1)
364         a = 1.0 / p
365
366         s = 0.0
367         i = k
368         while(i<=mm):
369             p = decay.lprod(lam, k, mm, 3, i)
370             s = s + (1 / lam[i]) * numpy.exp(-lam[i] * t) * (1.0 / p)
371             i = i + 1
372             s = s * (-1)**(mm-k+1)
373         return a + s
374
375     @staticmethod
376     def g_stable(t, k, m, lam):
377         mm = m - 1
378
379         p = decay.lprod(lam, k, mm, 1)
380         a = (t/p)
381
382         s = 0.0
383         i = k

```

```

384     while(i<=mm):
385         p = decay.lprod(lam, k, mm, 4, i)
386         s = s + p
387         i = i + 1
388     q = decay.lprod(lam, k, mm, 2)
389     b = (-s / q)
390
391     c = 0.0
392     i = k
393     while(i<= mm):
394         p = lam[i]**2 * decay.lprod(lam, k, mm, 3, i)
395         c = c + numpy.exp(-1.0 * lam[i] * t) / p
396         i = i + 1
397     c = c * (-1.0)**(mm - k)
398     nd = a + b + c
399
400     return nd
401
402
403 @staticmethod
404 def r(k, m, lam, b):
405     if(k == m):
406         return 1
407     else:
408         p = 1.0
409         i = k
410         while(i<=(m-1)):
411             p = p * b[i] * lam[i]
412             i = i + 1
413         return p
414
415 @staticmethod
416 def lprod(lam, k, m, t=1, i=None):
417     # PROD k,m lam[j]
418     if(t == 1):
419         p = 1.0
420         j = k
421         while(j<=m):
422             p = p * lam[j]
423             j = j + 1
424         return p
425     # PROD k,m lam[j]**2
426     elif(t == 2):
427         p = 1.0
428         j = k
429         while(j<=m):
430             p = p * lam[j]**2
431             j = j + 1
432         return p
433     # PROD k,m, i!=j (lam[i]-lam[j])
434     elif(t == 3):
435         p = 1.0
436         j = k
437         while(j<=m):
438             if(i != j):
439                 p = p * (lam[i] - lam[j])
440                 j = j + 1
441         return p
442     # PROD k,m, i!=j lam[j]
443     elif(t == 4):
444         p = 1.0

```

```

445     j = k
446     while(j<=m):
447         if(i != j):
448             p = p * lam[j]
449         j = j + 1
450     return p
451
452 @staticmethod
453 def pad(inp, l=16):
454     inp = str(inp)
455     while(len(inp)<l):
456         inp = inp + " "
457     return inp
458
459 @staticmethod
460 def hr(l=16):
461     inp = ""
462     while(len(inp)<l):
463         inp = inp + "="
464     return inp
465
466 @staticmethod
467 def get_file_dir(file_path):
468     file_path = file_path.strip()
469     if(file_path[0] != "/"):
470         root = os.getcwd()
471         file_path = root + "/" + file_path
472     file_path = file_path.split("/")
473     path = ""
474     for i in range(1,len(file_path) - 1):
475         path = path + "/" + file_path[i]
476     return path
477
478 @staticmethod
479 def make_dir(dir):
480     dirs = dir.split("/")
481     try:
482         dir = ''
483         for i in range(len(dirs)):
484             dir = dir + dirs[i]
485             if(not os.path.exists(dir) and dir.strip() != ''):
486                 os.mkdir(dir)
487             dir = dir + '/'
488     return True
489 except:
490     return False
491
492
493 @staticmethod
494 def test():
495     print("Test Decay")
496
497     # Load isotopes dictionary
498     decay.set("../data/isotopes.pz")
499
500     idata = {}
501     parent = 84216
502     time = 10
503     idata[84216] = {'w': 0.20, 'n0': 100.0}
504     idata[82212] = {'w': 0.0, 'n0': 5.0}
505     idata[83212] = {'w': 0.07, 'n0': 15.0}

```

```
506     idata[81208] = {'w': 0.005, 'n0': 0.0}
507     # 84Po212 0 0 (default)
508     idata[82208] = {'w': 0.01, 'n0': 300.0}
509     decay.calculate(parent, time, idata, "testing/log_84216_new.txt")
510
511
512 def main():
513     decay.test()
514
515 if __name__ == "__main__":
516     main()
```

B.3 Activity Equation Testing

B.3.1 Numeric Fortran Code

A simple Fortran code was created for each decay path used to test the activity equation and computer code. The version of the code used to calculate the amount of each isotope for the Po-216 decay chain is given in fig. B.3, and this code was modified for each decay chain tested.

Listing B.2: Bash file to compile Fortran test code

```

1 #!/bin/bash
2 mpif90 -g \
3 -fopenmp \
4 -fbounds-check \
5 -mtune=native \
6 kinds.f90 \
7 main.f90 \
8 -o test.x \
9 && ./test.x

```

Listing B.3: Fortran code to numerically estimate the amount of each isotope in the Polonium-216 decay chain

```

1 PROGRAM main_test
2 ! University of Birmingham
3 ! Ben Palmer
4 Use kinds
5 Use MPI
6
7 IMPLICIT NONE
8
9 CALL main()
10
11 CONTAINS
12
13 ! Subroutines
14
15
16 SUBROUTINE main()
17 !#####
18 ! PRIVATE VARIABLES
19 INTEGER(kind=StandardInteger) :: i, j, k
20 INTEGER(kind=StandardInteger) :: steps, chn
21 REAL(kind=DoubleReal) :: w(1:20)
22 REAL(kind=DoubleReal) :: t(1:20)
23 REAL(kind=DoubleReal) :: l(1:20)
24 REAL(kind=DoubleReal) :: n0(1:20)
25 REAL(kind=DoubleReal) :: n(1:20)
26 REAL(kind=DoubleReal) :: source(1:20)
27 REAL(kind=DoubleReal) :: loss(1:20)
28 REAL(kind=DoubleReal) :: t_end, t_step
29 !#####
30
31 print *, "Decay Test - Po216"
32
33 w(:) = 0.0D0
34 l(:) = 0.0D0
35 t(:) = 0.0D0
36 n0(:) = 0.0D0
37 source(:) = 0.0D0
38 loss(:) = 0.0D0
39 chn = 6

```

```

40
41 w(:) = 0.0D0
42 w(1) = 0.2D0
43 w(3) = 0.07D0
44 w(4) = 0.005D0
45 w(6) = 0.01D0
46
47 l(1) = 4.683427D+00 ! 84Po216
48 l(2) = 1.809595D-05 ! 82Pb212
49 l(3) = 1.908235D-04 ! 83Bi212
50 l(4) = 3.777781D-03 ! 81Tl208
51 l(5) = 2.310491D+06 ! 84Po212
52 ! 82Pb208
53
54 n0(:) = 0.0D0
55 n0(1) = 1.0D2
56 n0(2) = 5.0D0
57 n0(3) = 1.5D1
58 n0(6) = 3.0D2
59
60 n(:) = n0(:)
61
62 t_end = 10.0D0
63 t_step = t_end / 1.0D10
64
65 Do k=1,10
66 DO i=1,1000000000
67   source = 0.0D0
68   loss = 0.0D0
69
70   source(1) = t_step * w(1)
71   loss(1) = n(1) * (1.0D0 - exp(-l(1) * t_step))
72
73   source(2) = t_step * w(2) + loss(1)
74   loss(2) = n(2) * (1.0D0 - exp(-l(2) * t_step))
75
76   source(3) = t_step * w(3) + loss(2)
77   loss(3) = n(3) * (1.0D0 - exp(-l(3) * t_step))
78
79   source(4) = t_step * w(4) + 0.359300D0 * loss(3)
80   loss(4) = n(4) * (1.0D0 - exp(-l(4) * t_step))
81
82   source(5) = t_step * w(5) + 0.640700D0 * loss(3)
83   loss(5) = n(5) * (1.0D0 - exp(-l(5) * t_step))
84
85   source(6) = t_step * w(6) + loss(4) + loss(5)
86
87   n(1:chn) = n(1:chn) + source(1:chn) - loss(1:chn)
88
89 END DO
90 END DO
91
92
93 print *, "Po216 ", n(1)
94 print *, "Pb212 ", n(2)
95 print *, "Bi212 ", n(3)
96 print *, "Tl208 ", n(4)
97 print *, "Po212 ", n(5)
98 print *, "Pb208 ", n(6)
99
100

```

```
101 END SUBROUTINE main
102
103
104
105 END PROGRAM main_test
```

B.3.2 Chromium-49

Chromium-49 has a straightforward decay path, via positron emission (or electron capture) through to Vanadium-49 then to the stable isotope Titanium-49. Calculated isotope amounts are given for 5 different trial source rates and starting amounts; the results are listed in table B.2. The RSS error between the numeric and analytic code in this decay chain over all five trials was 6.52×10^{-3} and the total percentage error between the two was $2.17 \times 10^{-3}\%$.

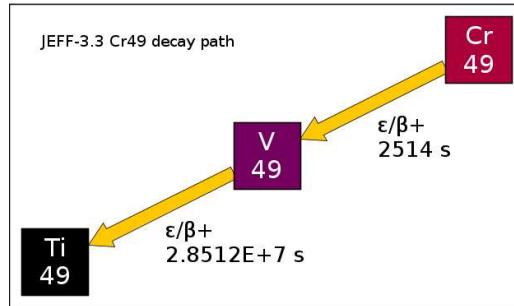


Figure B.1: Decay path for Chromium-49 [26]

		Trial 1	Trial 2	Trial 3	Trial 4	Trial 5
$^{49}_{24}Cr$	ω	0.0×10^0	1.0×10^2	0.0×10^0	2.5×10^0	2.5×10^0
	n_{start}	0.0×10^0	0.0×10^0	1.0×10^2	3.0×10^4	3.0×10^4
	Numeric n_{end}	0.0×10^0	3.62694×10^5	4.51169×10^{-9}	9.06734×10^3	9.06734×10^3
	Analytic n_{end}	0.0×10^0	3.62694×10^5	4.51169×10^{-9}	9.06734×10^3	9.06734×10^3
$^{49}_{23}V$	ω	0.0×10^0	0.0×10^0	0.0×10^0	0.0×10^0	1.07×10^0
	n_{start}	0.0×10^0	0.0×10^0	0.0×10^0	0.0×10^0	1.0×10^3
	Numeric n_{end}	0.0×10^0	8.26897×10^6	9.97990×10^1	2.36664×10^5	3.30013×10^5
	Analytic n_{end}	0.0×10^0	8.26897×10^6	9.97990×10^1	2.36664×10^5	3.30013×10^5
$^{49}_{22}Ti$	ω	0.0×10^0	0.0×10^0	0.0×10^0	0.0×10^0	3.2×10^0
	n_{start}	0.0×10^0	0.0×10^0	0.0×10^0	0.0×10^0	2.0×10^4
	Numeric n_{end}	0.0×10^0	8.33841×10^3	2.01023×10^{-1}	2.68767×10^2	2.96848×10^5
	Analytic n_{end}	0.0×10^0	8.33847×10^3	2.01025×10^{-1}	2.68769×10^2	2.96848×10^5

Table B.2: Chromium-49 - numeric vs analytic calculated radioactivity

B.3.3 Nickel-66

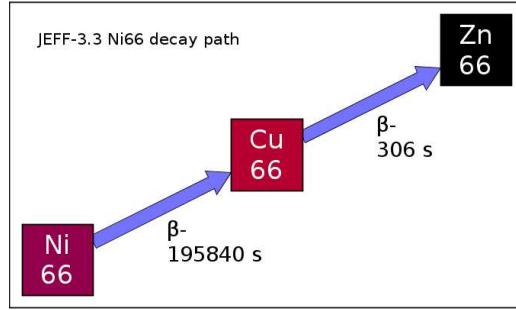


Figure B.2: Decay path for Nickel-66 [26]

Nickel-66 beta decays to Copper-66 and then to Zinc-66. The trial data are in table B.4. The RSS error between the numeric and analytic code in this decay chain over all five trials was 9.40×10^{-2} and the total percentage error between the two was $1.52 \times 10^{-4}\%$.

		Trial 1	Trial 2	Trial 3	Trial 4	Trial 5
$^{66}_{28}Ni$	ω	0.0×10^0	1.0×10^2	0.0×10^0	2.5×10^0	2.5×10^0
	n_{start}	0.0×10^0	0.0×10^0	1.0×10^2	3.0×10^4	3.0×10^4
Numeric	n_{end}	0.0×10^0	7.44391×10^6	7.36534×10^1	2.08194×10^5	2.08194×10^5
Analytic	n_{end}	0.0×10^0	7.44391×10^6	7.36534×10^1	2.08194×10^5	2.08194×10^5
$^{66}_{29}Cu$	ω	0.0×10^0	0.0×10^0	0.0×10^0	0.0×10^0	1.07×10^0
	n_{start}	0.0×10^0	0.0×10^0	0.0×10^0	0.0×10^0	1.0×10^3
Numeric	n_{end}	0.0×10^0	1.15802×10^4	1.15264×10^{-1}	3.24085×10^2	7.96452×10^2
Analytic	n_{end}	0.0×10^0	1.15802×10^4	1.15264×10^{-1}	3.24085×10^2	7.96452×10^2
$^{49}_{22}Ti$	ω	0.0×10^0	0.0×10^0	0.0×10^0	0.0×10^0	3.2×10^0
	n_{start}	0.0×10^0	0.0×10^0	0.0×10^0	0.0×10^0	2.0×10^4
Numeric	n_{end}	0.0×10^0	1.18451×10^6	2.62314×10^1	3.74822×10^4	4.26938×10^5
Analytic	n_{end}	0.0×10^0	1.18451×10^6	2.62314×10^1	3.74822×10^4	4.26938×10^5

Table B.4: Nickel-66 - numeric vs analytic calculated radioactivity

B.3.4 Caesium-125

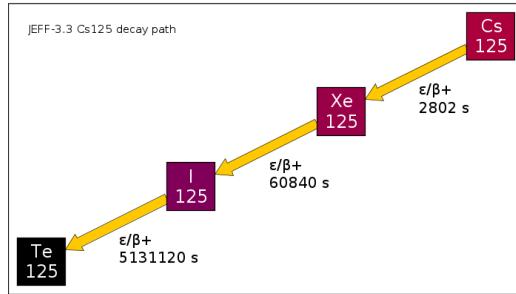


Figure B.3: Decay path for Caesium-125 [26]

The RSS error between the numeric and analytic code in this decay chain over all five trials was 6.42×10^{-1} and the total percentage error between the two was $1.23 \times 10^{-3}\%$.

		Trial 1	Trial 2	Trial 3	Trial 4	Trial 5
$^{125}_{55}Cs$	ω	0.0×10^0	1.0×10^2	0.0×10^0	5.0×10^1	2.3×10^1
	n_{start}	0.0×10^0	0.0×10^0	1.0×10^2	2.0×10^5	3.0×10^5
Numeric	n_{end}	0.0×10^0	4.0424×10^5	5.2204×10^{-8}	2.0212×10^5	9.2976×10^4
Analytic	n_{end}	0.0×10^0	4.0424×10^5	5.2204×10^{-8}	2.0212×10^5	9.2976×10^4
$^{125}_{54}Xe$	ω	0.0×10^0	0.0×10^0	0.0×10^0	0.0×10^0	5.4×10^1
	n_{start}	0.0×10^0	0.0×10^0	0.0×10^0	0.0×10^0	2.0×10^2
Numeric	n_{end}	0.0×10^0	5.3391×10^6	3.9172×10^1	2.7479×10^6	4.3142×10^6
Analytic	n_{end}	0.0×10^0	5.3391×10^6	3.9172×10^1	2.7479×10^6	4.3142×10^6
$^{125}_{53}I$	ω	0.0×10^0	0.0×10^0	0.0×10^0	0.0×10^0	2.1×10^1
	n_{start}	0.0×10^0	0.0×10^0	0.0×10^0	0.0×10^0	1.7×10^6
Numeric	n_{end}	0.0×10^0	2.8851×10^6	6.0438×10^1	1.5634×10^6	6.0191×10^6
Analytic	n_{end}	0.0×10^0	2.8851×10^6	6.0438×10^1	1.5634×10^6	6.0191×10^6
$^{125}_{52}Te$	ω	0.0×10^0	0.0×10^0	0.0×10^0	0.0×10^0	4.0×10^0
	n_{start}	0.0×10^0	0.0×10^0	0.0×10^0	0.0×10^0	5.0×10^4
Numeric	n_{end}	0.0×10^0	1.1544×10^4	3.8975×10^{-1}	6.5513×10^3	4.3678×10^5
Analytic	n_{end}	0.0×10^0	1.1544×10^4	3.8975×10^{-1}	6.5513×10^3	4.3678×10^5

Table B.6: Caesium-125 - numeric vs analytic calculated radioactivity

B.3.5 Bismuth-213

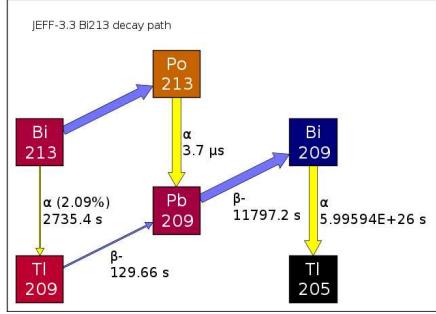


Figure B.4: Decay path for Bismuth-213 [26]

The previous decay examples have been straight chains with no branches, but Bismuth-213 is the first discussed in this section to have a branch. There's approximately an 98% chance that $^{213}_{83}Bi$ will decay to $^{213}_{83}Tl$ by alpha decay and just over 2% chance that it will decay to $^{213}_{84}Po$ by beta decay. These both go on to decay to $^{209}_{82}Pb$ that then undergoes beta decay to $^{209}_{83}Bi$. There is another step in the chain (to the stable isotope $^{205}_{81}Tl$), but $^{209}_{83}Bi$ is observationally stable with an extremely long half life and the decay code terminates the chain at this point.

The RSS error between the numeric and analytic code in this decay chain over all five trials was 2.03×10^{13} . This is large, but due mostly to the larger amounts used in this example. The total percentage error between the two was $6.07 \times 10^{3}\%$ which is again large, but this is due to the numeric code and the short half life of Po-213 in comparison to the time step used. Without the contribution of Po-213 the total percentage error between the two sets of results is just $2.37 \times 10^{-4}\%$.

		Trial 1	Trial 2	Trial 3	Trial 4	Trial 5
$^{213}_{83}Bi$	ω	0.0×10^0	1.0×10^2	0.0×10^0	3.0×10^5	3.0×10^5
	n_{start}	0.0×10^0	0.0×10^0	1.0×10^0	1.0×10^{10}	1.0×10^{10}
	Numeric	n_{end}	0.0×10^0	3.9463×10^5	3.1024×10^{-8}	1.1839×10^9
	Analytic	n_{end}	0.0×10^0	3.9463×10^5	3.1024×10^{-8}	1.1839×10^9
$^{213}_{84}Po$	ω	0.0×10^0	0.0×10^0	0.0×10^0	0.0×10^0	7.0×10^3
	n_{start}	0.0×10^0	0.0×10^0	0.0×10^0	0.0×10^0	2.0×10^7
	Numeric	n_{end}	0.0×10^0	8.4594×10^{-3}	6.6504×10^{-16}	2.5378×10^1
	Analytic	n_{end}	0.0×10^0	5.2264×10^{-4}	4.1088×10^{-17}	1.5679×10^0
$^{209}_{81}Tl$	ω	0.0×10^0	0.0×10^0	0.0×10^0	0.0×10^0	2.0×10^4
	n_{start}	0.0×10^0	0.0×10^0	0.0×10^0	0.0×10^0	2.0×10^6
	Numeric	n_{end}	0.0×10^0	3.9096×10^2	3.2265×10^{-11}	1.1729×10^6
	Analytic	n_{end}	0.0×10^0	3.9096×10^2	3.2265×10^{-11}	1.1729×10^6
$^{209}_{82}Pb$	ω	0.0×10^0	0.0×10^0	0.0×10^0	0.0×10^0	1.0×10^0
	n_{start}	0.0×10^0	0.0×10^0	0.0×10^0	0.0×10^0	1.0×10^7
	Numeric	n_{end}	0.0×10^0	1.6881×10^6	8.1281×10^{-1}	5.1457×10^9
	Analytic	n_{end}	0.0×10^0	1.6881×10^6	8.1281×10^{-1}	5.1457×10^9
$^{209}_{83}Bi$	ω	0.0×10^0	0.0×10^0	0.0×10^0	0.0×10^0	3.5×10^8
	n_{start}	0.0×10^0	0.0×10^0	0.0×10^0	0.0×10^0	1.0×10^6
	Numeric	n_{end}	0.0×10^0	6.5568×10^6	9.9187×10^1	2.9589×10^{10}
	Analytic	n_{end}	0.0×10^0	6.5568×10^6	9.9187×10^1	2.9589×10^{10}
$^{205}_{81}Tl$	ω	0.0×10^0	0.0×10^0	0.0×10^0	0.0×10^0	1.0×10^{-10}
	n_{start}	0.0×10^0	0.0×10^0	0.0×10^0	0.0×10^0	1.0×10^{-10}
	Numeric	n_{end}	0.0×10^0	0.0×10^0	0.0×10^0	0.0×10^0
	Analytic	n_{end}	0.0×10^0	0.0×10^0	0.0×10^0	0.0×10^0

Table B.8: Bismuth-213 - numeric vs analytic calculated radioactivity

B.3.6 Polonium-216

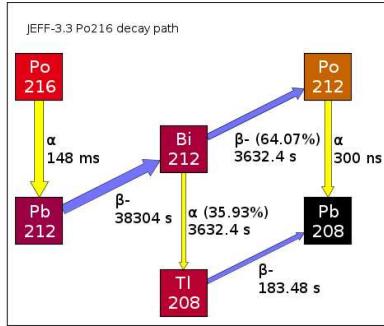


Figure B.5: Decay path for Bismuth-213 [26]

The RSS error between the numeric and analytic code in this decay chain over all five trials was 3.85×10^1 and the total percentage error between the two was $7.94 \times 10^4\%$. This large error was due to a short half life isotope (Po-212) and the comparatively long time step in the numeric solver. Without this result, the agreement between numeric and analytic is $6.10 \times 10^{-2}\%$.

		Trial 1	Trial 2	Trial 3	Trial 4	Trial 5
$^{216}_{84}Po$	ω	0.0×10^0	1.0×10^2	0.0×10^0	3.5×10^2	2.0×10^{-1}
	n_{start}	0.0×10^0	0.0×10^0	1.0×10^2	5.0×10^4	1.0×10^2
	n_{end}	0.0×10^0	2.13562×10^1	0.0×10^0	7.47467×10^1	4.27124×10^{-2}
	Analytic	0.0×10^0	2.13519×10^1	0.0×10^0	7.47316×10^1	4.27038×10^{-2}
$^{212}_{82}Pb$	ω	0.0×10^0	0.0×10^0	0.0×10^0	0.0×10^0	2.0×10^{-1}
	n_{start}	0.0×10^0	0.0×10^0	0.0×10^0	0.0×10^0	5.0×10^0
	Numeric	0.0×10^0	4.36891×10^6	2.09405×10^1	1.53016×10^7	1.31287×10^4
	Analytic	0.0×10^0	4.36891×10^6	2.09405×10^1	1.53017×10^7	1.31287×10^4
$^{212}_{83}Bi$	ω	0.0×10^0	0.0×10^0	0.0×10^0	0.0×10^0	7.0×10^{-2}
	n_{start}	0.0×10^0	0.0×10^0	0.0×10^0	0.0×10^0	1.50×10^1
	Numeric	0.0×10^0	4.02810×10^5	2.19385×10^0	1.41093×10^6	1.57757×10^3
	Analytic	0.0×10^0	4.02810×10^5	2.19385×10^0	1.41093×10^6	1.57757×10^3
$^{208}_{81}Tl$	ω	0.0×10^0	0.0×10^0	0.0×10^0	0.0×10^0	5.0×10^{-3}
	n_{start}	0.0×10^0	0.0×10^0	0.0×10^0	0.0×10^0	1.0×10^1
	Numeric	0.0×10^0	7.30001×10^3	4.00077×10^{-2}	2.55700×10^4	1.83923×10^{-5}
	Analytic	0.0×10^0	$\times 10$	$\times 10$	$\times 10$	$\times 10$
$^{212}_{84}Po$	ω	0.0×10^0	0.0×10^0	0.0×10^0	0.0×10^0	2.0×10^{-2}
	n_{start}	0.0×10^0	0.0×10^0	0.0×10^0	0.0×10^0	1.7×10^1
	Numeric	0.0×10^0	4.25501×10^{-3}	2.31743×10^{-8}	1.49041×10^{-2}	1.83923×10^{-5}
	Analytic	0.0×10^0	2.13149×10^{-5}	1.16088×10^{-10}	7.46601×10^{-5}	9.21338×10^{-8}
$^{208}_{82}Pb$	ω	0.0×10^0	0.0×10^0	0.0×10^0	0.0×10^0	1.0×10^{-2}
	n_{start}	0.0×10^0	0.0×10^0	0.0×10^0	0.0×10^0	3.0×10^2
	Numeric	0.0×10^0	3.86096×10^6	7.68257×10^1	1.35518×10^7	2.07028×10^4
	Analytic	0.0×10^0	3.86096×10^6	7.68257×10^1	1.35518×10^7	2.07028×10^4

Table B.10: Polonium-216 - numeric vs analytic calculated radioactivity

B.3.7 Radon-218

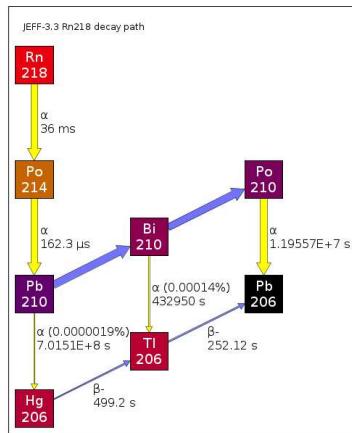


Figure B.6: Decay path for Bismuth-213 [26]

The RSS error between the numeric and analytic code in this decay chain over all five trials was 1.73×10^3 and the total percentage error between the two was $5.93 \times 10^1\%$. However, if the short half life isotope Po-214 is excluded, the error is just $5.15 \times 10^{-1}\%$.

		Trial 1	Trial 2	Trial 3	Trial 4	Trial 5
$^{218}_{86}Rn$	ω	0.0×10^0	1.0000×10^0	0.0000×10^0	3.5×10^2	3.5000×10^2
	n_{start}	0.0000×10^0	0.0000×10^0	1.0000×10^0	5.0000×10^4	5.0000×10^4
	Numeric	0.0000×10^0	5.1980×10^0	0.0000×10^0	1.8193×10^1	1.8193×10^1
	Analytic	0.0000×10^0	5.1937×10^0	0.0000×10^0	1.8178×10^1	1.8178×10^1
$^{214}_{84}Po$	ω	0.0000×10^0	0.0000×10^0	0.0000×10^0	0.0000×10^0	7.0000×10^1
	n_{start}	0.0000×10^0	0.0000×10^0	0.0000×10^0	0.0000×10^0	1.0000×10^3
	Numeric	0.0000×10^0	2.8000×10^{-2}	0.0000×10^0	9.8000×10^{-2}	1.1760×10^{-1}
	Analytic	0.0000×10^0	2.3415×10^{-2}	0.0000×10^0	8.1952×10^{-2}	9.8343×10^{-1}
$^{210}_{82}Pb$	ω	0.0000×10^0	0.0000×10^0	0.0000×10^0	0.0000×10^0	1.0000×10^3
	n_{start}	0.0000×10^0	0.0000×10^0	0.0000×10^0	0.0000×10^0	1.2000×10^2
	Numeric	0.0000×10^0	8.6396×10^6	9.9991×10^1	3.0289×10^7	1.2273×10^8
	Analytic	0.00000×10^0	8.6396×10^6	9.9991×10^1	3.0289×10^7	1.2273×10^8
$^{206}_{80}Hg$	ω	0.0000×10^0	0.0000×10^0	0.0000×10^0	0.0000×10^0	1.7000×10^1
	n_{start}	0.0000×10^0	0.0000×10^0	0.0000×10^0	0.0000×10^0	7.0000×10^2
	Numeric	0.0000×10^0	1.1585×10^{-7}	1.3520×10^{-12}	4.0614×10^{-7}	1.2243×10^4
	Analytic	0.00000×10^0	1.1584×10^{-7}	1.3519×10^{-12}	4.0611×10^{-7}	1.2243×10^4
$^{210}_{83}Bi$	ω	0.0000×10^0	0.0000×10^0	0.0000×10^0	0.0000×10^0	2.3000×10^1
	n_{start}	0.0000×10^0	0.0000×10^0	0.0000×10^0	0.0000×10^0	4.1000×10^1
	Numeric	0.0000×10^0	3.5238×10^2	7.9731×10^{-3}	1.2373×10^3	1.8609×10^{-6}
	Analytic	0.00000×10^0	3.5236×10^2	7.9725×10^{-3}	1.2372×10^3	1.8609×10^6
$^{206}_{81}Tl$	ω	0.0000×10^0	0.0000×10^0	0.0000×10^0	0.0000×10^0	8.1000×10^1
	n_{start}	0.0000×10^0	0.0000×10^0	0.0000×10^0	0.0000×10^0	5.0000×10^2
	Numeric	0.0000×10^0	3.4319×10^{-7}	7.1575×10^{-12}	1.2047×10^{-6}	3.5646×10^4
	Analytic	0.00000×10^0	3.4316×10^{-7}	7.1570×10^{-12}	1.2046×10^{-6}	3.5646×10^4
$^{210}_{84}Po$	ω	0.0000×10^0	0.0000×10^0	0.0000×10^0	0.0000×10^0	5.2000×10^2
	n_{start}	0.0000×10^0	0.0000×10^0	0.0000×10^0	0.0000×10^0	1.5000×10^3
	Numeric	0.0000×10^0	1.6413×10^1	5.6320×10^{-4}	5.7726×10^1	4.4948×10^7
	Analytic	0.00000×10^0	1.6411×10^1	5.6316×10^{-4}	5.7722×10^1	4.4948×10^7
$^{206}_{82}Pb$	ω	0.0000×10^0	0.0000×10^0	0.0000×10^0	0.0000×10^0	1.0000×10^3
	n_{start}	0.0000×10^0	0.0000×10^0	0.0000×10^0	0.0000×10^0	1.0000×10^4
	Numeric	0.0000×10^0	2.0729×10^{-2}	9.5248×10^{-7}	7.3026×10^{-2}	9.4943×10^7
	Analytic	0.00000×10^0	2.0747×10^{-2}	9.5242×10^{-7}	7.3089×10^{-2}	9.4943×10^7

Table B.12: Radon-218 - numeric vs analytic calculated radioactivity

Appendix C

Activity Code

C.1 Activity V1

C.1.1 Paper published in Computer Physics Communications

The Activity code was published in the Computer Physics Communications journal and this is included in the following pages. It was submitted in 2015 and published in 2017.

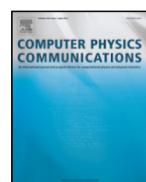
Computer Physics Communications 216 (2017) 138–144



Contents lists available at ScienceDirect

Computer Physics Communications

journal homepage: www.elsevier.com/locate/cpc



Activity computer program for calculating ion irradiation activation[☆]

Ben Palmer ^{*}, Brian Connolly, Mark Read

University of Birmingham, United Kingdom



ARTICLE INFO

Article history:

Received 1 June 2015

Received in revised form 15 February 2017

Accepted 19 February 2017

Available online 29 March 2017

ABSTRACT

A computer program, Activity, was developed to predict the activity and gamma lines of materials irradiated with an ion beam. It uses the TENDL (Koning and Rochman, 2012) [1] proton reaction cross section database, the Stopping and Range of Ions in Matter (SRIM) (Biersack et al., 2010) code, a Nuclear Data Services (NDS) radioactive decay database (Sonzogni, 2006) [2] and an ENDF gamma decay database (Herman and Chadwick, 2006) [3]. An extended version of Bateman's equation is used to calculate the activity at time t, and this equation is solved analytically, with the option to also solve by numeric inverse Laplace Transform as a failsafe. The program outputs the expected activity and gamma lines of the activated material.

Program summary

Program title: Activity

Catalogue identifier: AFBS_v1_0

Program summary URL: http://cpc.cs.qub.ac.uk/summaries/AFBS_v1_0.html

Program obtainable from: CPC Program Library, Queen's University, Belfast, N. Ireland

Licensing provisions: GNU GPL v3

No. of lines in distributed program, including test data, etc.: 688828

No. of bytes in distributed program, including test data, etc.: 71056048

Distribution format: tar.gz

Programming language: Fortran.

Computer: PCs or HPCs.

Operating system: Linux (tested on Debian).

Has the code been vectorized or parallelized?: OpenMPI

RAM: 250MB per process + 200MB overhead

Classification: 2.2, 17.8.

Nature of problem: To calculate the predicted activity of an ion irradiated target. The expected range of ion energies is between 1MeV and 200MeV; this is the range of the available ion cross section data.

Solution method: The program loads cross section data from the TENDL database and trajectory data from a SRIM [1] simulation exyz data file. It uses this data to calculate the production/loss rate of each isotope in the simulated target. Radioactive decay equations are used to calculate the amounts and activity of each radioactive isotope at the set time.

Running time: Typically the Activity program runs each input from seconds to no more than several minutes.

References:

- [1] SRIM – The stopping and range of ions in matter (2010). Ziegler, James F., Ziegler, M.D. and Biersack, J.P. 2010, Nuclear Instruments and Methods in Physics Research Section B, Vol. 268, pp. 1818–1823.

© 2017 Elsevier B.V. All rights reserved.

[☆] This paper and its associated computer program are available via the Computer Physics Communication homepage on ScienceDirect (<http://www.sciencedirect.com/science/journal/00104655>).

* Corresponding author.

E-mail address: benpalmer1983@gmail.com (B. Palmer).

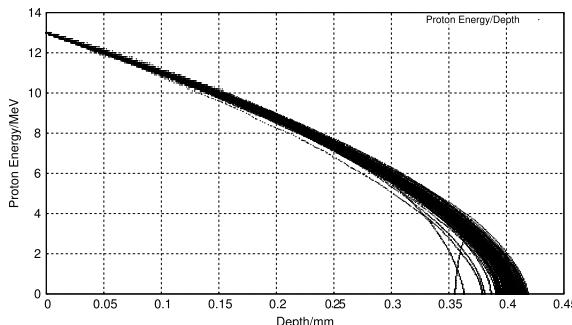


Fig. 1. One hundred simulated 13 MeV proton energy loss curves in Fe simulated with SRIM [4].

1. Background

1.1. Ion irradiation at the University of Birmingham

The Scanditronix MC-40 Cyclotron is used at the University of Birmingham to create a beam of protons or other light ions. The energies of these ions are typically between 10 MeV and 60 MeV with beam currents ranging up to 50 μA (3.1×10^{14} protons per second). Target materials are irradiated by this cyclotron for a number of reasons, including purposely creating radioactive isotopes for the nearby Queen Elizabeth Hospital, investigating ion irradiation damage and emulating neutron irradiation.

The cyclotron is usually used to create radioactive isotopes for medical use, but an additional beam line has been devoted to material science investigations into radiation damage. While the creation of radioactive isotopes is desired in some cases, material being tested for radiation damage should preferably have low levels of radioactivity.

It is expensive to arrange the irradiation of target materials by high energy neutrons sources, whereas it is relatively inexpensive to irradiate using an ion beam on the MC-40 Cyclotron. The energies can be controlled, and a set dose at a single energy, or a range of energies, can be precisely deposited into the target material.

The Activity code discussed here was developed to calculate the activity of a target material irradiated by a proton beam. It has been developed in Fortran and uses data from the TENDL-2013 proton cross section database, SRIM ion transport code and NDS radioactive decay database.

1.2. Simulating ion irradiation with SRIM

A package of ion transport codes, SRIM, is freely available to download and use to investigate the transport of ions through matter. SRIM uses the binary collision approximation (BCA) to simulate the passage of ions in a material. It is an approximate method, and one key restriction is that it does not take into account the structure of the material, and this approximation is therefore also imposed on the Activity code.

One file that SRIM creates is of importance to the Activity code, and that is the trajectory file that contains the energy and x , y , z co-ordinate data points for simulated ions moving through matter. Fig. 1 shows the trajectory of one hundred 13 MeV protons entering and passing through an Iron target, and it is this set of data points (together with the cross section database) that the Activity code uses to calculate the reaction rates for the transmutation of nuclei in the target. At higher energies, the ions slow as they lose energy due to electronic stopping, but as the ion energy drops the mechanism of loss through nuclear collisions becomes important.

The spreading of ion depths at lower energies is a result of the higher momentum transfer during nuclear collisions, as can be seen in Fig. 1.

1.3. Transmutation of nuclei by ion irradiation

Considering a simplified nuclear potential well, energetic protons approaching a nucleus may overcome the Coulomb potential barrier. They are captured by the nucleus and held within the potential well by the strong nuclear force. This process may leave the nucleus in an excited and unstable state, depending on the input energy of the proton and configuration of nucleons. The process is probabilistic, and the average chance of a reaction (the microscopic cross section) may be measured as a function of the projectile, projectile energy and target, either experimentally or by optical model potential calculations. The reaction rate is calculated from the microscopic cross section using the following equation:

$$R = \frac{J}{e} n_t \sigma \cdot 10^{-28} \delta t \quad (1)$$

- R Reaction Rate (reactions per second)
- J Beam current (A)
- n_t Number density of target (atoms per cubic meter)
- σ Microscopic reaction cross section (barns)
- e Elementary charge (1.602177E-19C)
- δt Target thickness (m).

1.4. Radioactive decay

Radioactive decay is the random change in nucleons or energy state of an unstable nucleus. It is impossible to predict when a single nucleus will decay, but the decay of a collection of nuclei is statistical in nature. The radioactivity and number of unstable nuclei at time t can be predicted using the decay constant, λ , for the radioactive isotope. This constant is defined as follows:

$$\lambda = -\frac{N'(t)}{N(t)}. \quad (2)$$

The number of radioactive nuclei $N(t)$ at time t is given by the following equation, where $N(0)$ is the starting number of nuclei:

$$N(t) = N(0) \exp(-t\lambda). \quad (3)$$

The activity $A(t)$ of the radioactive nuclei is predicted at time t by using the following equations, where $N'(t)$ is the change in amount of nuclei with respect to time:

$$A(t) = -N'(t) = \lambda N(t) \quad (4)$$

$$A(t) = \lambda N(0) \exp(-t\lambda). \quad (5)$$

1.5. Bateman equation for radioactive decay

The English mathematician Harry Bateman derived an Eq. (6) to calculate the amount of each isotope in a decay chain, illustrated in Fig. 2, at time t .

$$N_n(t) = \sum_{i=1}^{i=n} \left(\left(\prod_{j=i}^{j=n-1} \lambda_{(ij+1)} \right) \sum_{j=i}^{j=n} \left(\frac{N_{i0} \exp(-\lambda_j t)}{\prod_{p=i, p \neq j}^{p=n} (\lambda_p - \lambda_j)} \right) \right). \quad (6)$$

When a radioactive isotope decays, there may be more than one mode of decay, and this leads to branching factors. Pb-214 only decays via beta decay to Bi-214, giving a branching factor of 1.0, whereas Bi-214 has a 99.979% chance of decaying to Po-214 by beta decay and a 0.021% of emitting an alpha particle and decaying to Tl-210 (branching factors of 0.99979 and 0.00021 respectively) [5].

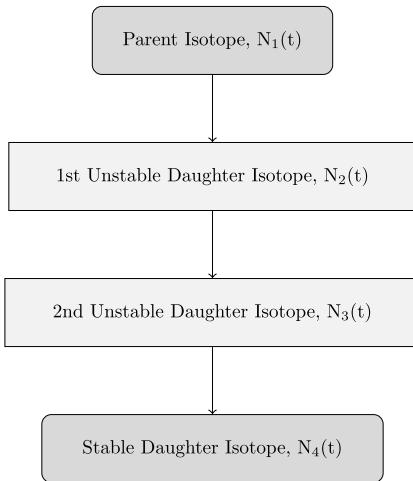


Fig. 2. An example decay chain from an unstable parent isotope, through unstable daughter isotopes ending with a stable daughter isotope.

When a target material is irradiated, there is a source term for transmuted nuclei due to the irradiation. The daughter isotopes of these transmuted isotopes will also be affected by the irradiation and will transmute further, giving a source term for each daughter isotope as a result of the irradiation. Sources for each isotope in the decay chain, and branching factors between a parent isotope and its daughter isotope/s must be accounted for.

The Bateman equation was derived using Laplace transforms, and this same method has been used to develop a modified equation that incorporates branching factors and production rates for each isotope in the decay chain, as illustrated by Fig. 3.

1.6. Laplace transform

Laplace Transforms (7) are a useful mathematical tool, and allow ordinary differential equations to be solved by simple algebraic manipulation in the s domain. Bateman took advantage of Laplace Transforms in deriving his equation, and this is the method that has been taken here as well.

$$F(s) = \int_0^\infty f(t) \exp(-st) dt. \quad (7)$$

1.7. Constructing the differential equations

The first step is to set up differential equations for the parent isotope, unstable daughter isotopes and stable daughter isotope. The parent isotope has a source term, due to production, and a loss term, due to decay. The unstable daughter isotopes have two source terms, from the production by irradiation induced transmutation and the decay of preceding isotopes in the decay chain, and a loss term, due to decay. Finally, the stable daughter that finalizes the decay chain has two source terms (the same as the unstable daughters) but no loss term.

The variables (and vectors) used in these equations are defined as follows:

- $\vec{\lambda}$ vector containing isotope decay constants λ_i
- \vec{b} vector containing isotope to isotope branching factors b_i
- \vec{w} vector containing isotope production rates w_i
- t time at which activity/amount of isotope is measured
- $N_i(0)$ starting amount of the ith isotope

- $N_i(t)$ amount of the ith isotope at time t
- $N'_i(t)$ change in amount of the ith isotope, with respect to time, at time t .

The differential equations for the parent isotope (first isotope), unstable daughter isotopes (ith isotopes) and stable, final, daughter isotope (zth isotope) in the time domain are as follows:

$$N'_1(t) = \omega_1 - \lambda_1 N_1(t) \quad (8)$$

$$N'_i(t) = \omega_i + b_{i-1} \lambda_{i-1} N_{i-1}(t) - \lambda_i N_i(t) \quad (9)$$

$$N'_z(t) = \omega_z + b_{z-1} \lambda_{z-1} N_{z-1}(t). \quad (10)$$

Applying the Laplace Transform to these three differential equations allows them to be manipulated and solved algebraically in the s-domain.

$$N_1(s) = \frac{1}{s + \lambda_1} N_1(0) + \frac{1}{s(s + \lambda_1)} \omega_1 \quad (11)$$

$$N_i(s) = \frac{1}{s(s + \lambda_i)} (\omega_i) + \frac{1}{s + \lambda_i} (b_{i-1} \lambda_{i-1} N_{i-1}(s)) \\ + \frac{1}{s + \lambda_i} N_i(0) \quad (12)$$

$$N_z(s) = \frac{1}{s^2} \omega_z + \frac{1}{s} b_{z-1} \lambda_{z-1} N_{z-1}(s) + \frac{1}{s} N_z(0). \quad (13)$$

1.8. Numerical inversion of the Laplace Transform

The Gaver–Stehfest [6] algorithm was developed in the 1960s and 1970s and is a method of calculating the inverse of a Laplace Transform in the real number domain. It is an easy to implement and reasonably accurate method, although it is an approximation to the real value. A comparison between an analytic and numeric inversion for the unstable isotope Po-218 is discussed at the end of this section (Fig. 4).

$$f(t) \approx f_n(t) = \frac{\ln(2)}{t} \sum_{k=1}^{2n} a_k(n) F(s) \text{ where } n \geq 1, t > 0 \quad (14)$$

$$s = \frac{k \ln(2)}{t} \quad (15)$$

$$a_k(n) = \frac{(-1)^{(n+k)}}{n!} \sum_{j=\text{Floor}(\frac{k+1}{2})} j^{n+1} \binom{n}{j} \binom{2j}{j} \binom{j}{k-j}. \quad (16)$$

The equation for the ith isotope may be calculated by recursively calculating the equations by numeric inversion, starting from the first (parent isotope) and inserting the result into each subsequent recursion until the ith isotope is reached (changing the equations appropriately for the parent, unstable daughter and stable daughter isotopes).

1.9. Analytic solution by partial fraction expansion

The equation for the ith isotope in the s domain can be written in full by substituting the preceding equation until the parent isotope is reached, and this full equation may be rearranged with the production amount of each isotope and starting amount of each isotope in individual terms. Each of these terms is multiplied by a fraction that can be expanded, using partial fractions, and inverted analytically.

This is illustrated with an example unstable isotope, fourth in the decay chain (including the parent isotope):

$$N_4(s) = \frac{1}{(s + \lambda_1)(s + \lambda_2)(s + \lambda_3)(s + \lambda_4)} b_2 b_3 b_4 \lambda_1 \lambda_2 \lambda_3 N_1(0)$$

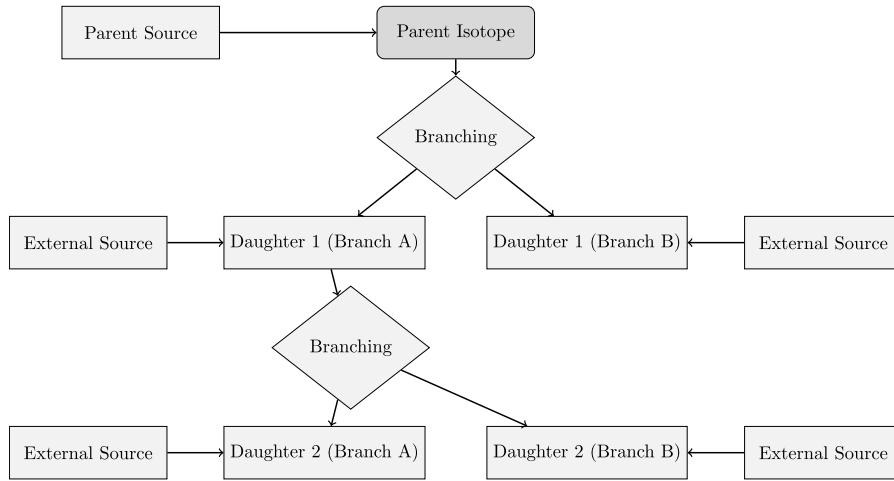


Fig. 3. An example of several decay chains including branching factors and possible external source terms for each isotope on each chain.

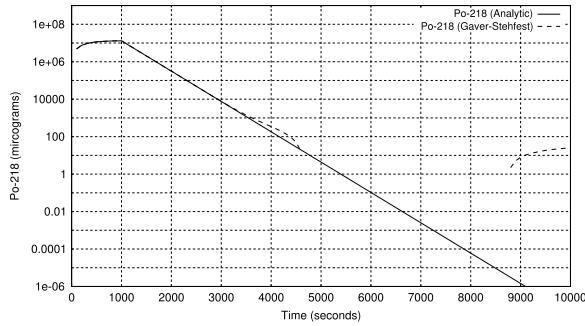


Fig. 4. Decay of Po-218: Analytic and Gaver–Stehfest Calculation [5].

$$\begin{aligned}
 & + \frac{1}{(s + \lambda_2)(s + \lambda_3)(s + \lambda_4)} b_3 b_4 \lambda_2 \lambda_3 N_2(0) \\
 & + \frac{1}{(s + \lambda_3)(s + \lambda_4)} b_4 \lambda_3 N_3(0) + \frac{1}{(s + \lambda_4)} N_4(0) \\
 & + \frac{1}{s(s + \lambda_1)(s + \lambda_2)(s + \lambda_3)(s + \lambda_4)} b_2 b_3 b_4 \lambda_1 \lambda_2 \lambda_3 \omega_1 \\
 & + \frac{1}{s(s + \lambda_2)(s + \lambda_3)(s + \lambda_4)} b_3 b_4 \lambda_2 \lambda_3 \omega_2 \\
 & + \frac{1}{s(s + \lambda_3)(s + \lambda_4)} b_4 \lambda_3 \omega_3 + \frac{1}{s(s + \lambda_4)} \omega_4. \quad (17)
 \end{aligned}$$

An example stable isotope, fourth (last) in the decay chain (including the parent isotope):

$$\begin{aligned}
 N_4(s) = & \frac{1}{s(s + \lambda_1)(s + \lambda_2)(s + \lambda_3)} b_2 b_3 b_4 \lambda_1 \lambda_2 \lambda_3 N_1(0) \\
 & + \frac{1}{s(s + \lambda_2)(s + \lambda_3)} b_3 b_4 \lambda_2 \lambda_3 N_2(0) \\
 & + \frac{1}{s(s + \lambda_3)} b_4 \lambda_3 N_3(0) + N_4(0) \\
 & + \frac{1}{s^2(s + \lambda_1)(s + \lambda_2)(s + \lambda_3)} b_2 b_3 b_4 \lambda_1 \lambda_2 \lambda_3 \omega_1 \\
 & + \frac{1}{s^2(s + \lambda_2)(s + \lambda_3)} b_3 b_4 \lambda_2 \lambda_3 \omega_2
 \end{aligned}$$

$$+ \frac{1}{s^2(s + \lambda_3)} b_4 \lambda_3 \omega_3 + \frac{1}{s^2} \omega_4. \quad (18)$$

By using partial fraction expansion and standard Laplace Transforms, the set of equations below is used to calculate the amount of the m th isotope in the decay chain, providing the m th isotope is unstable.

$$N_m(t; \vec{\lambda}, \vec{b}, \vec{w}) = \sum_{k=1,m} r(k; \vec{\lambda}, \vec{b}) [f(t; k, m, \vec{\lambda}) N_k(0) + g(t; k, m, \vec{\lambda}) w_k] \quad (19)$$

$$r(k, m, \vec{\lambda}) = \begin{cases} \prod_{i=k, m-1} (b_{i+1} \lambda_i), & \text{if } k < m \\ 1, & \text{if } k = m \end{cases} \quad (20)$$

$$f(t; k, m, \vec{\lambda}) = (-1)^{m-k} \sum_{i=k, m} \left[\exp(-\lambda_i t) \prod_{j=k, m; j \neq i} \left(\frac{1}{\lambda_i - \lambda_j} \right) \right] \quad (21)$$

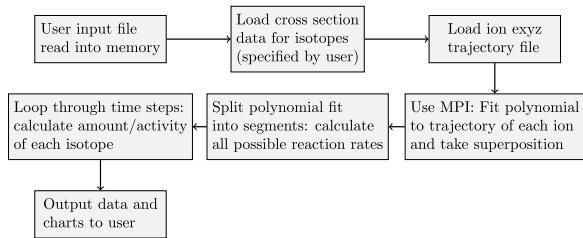
$$g(t; k, m, \vec{\lambda}) = \frac{1}{\prod_{i=k, m} \lambda_i} + (-1)^{m-k+1} \times \sum_{i=k, m} \left[\frac{1}{\lambda_i} \exp(-\lambda_i t) \prod_{j=k, m; j \neq i} \left(\frac{1}{\lambda_i - \lambda_j} \right) \right]. \quad (22)$$

The set of equations below is used to calculate the amount of the m th isotope in the decay chain, where the m th isotope is stable.

$$N_m(t; \vec{\lambda}, \vec{b}, \vec{w}) = N_m + w_m t + \sum_{k=1, m-1} r(k; \vec{\lambda}, \vec{b}) \times [f(t; k, m-1, \vec{\lambda}) N_k(0) + g(t; k, m, \vec{\lambda}) w_k] \quad (23)$$

$$r(k, m, \vec{\lambda}) = \begin{cases} \prod_{i=k, m-1} (b_{i+1} \lambda_i), & \text{if } k < m \\ 1, & \text{if } k = m \end{cases} \quad (24)$$

$$f(t; k, m, \vec{\lambda}) = \frac{1}{\prod_{i=k, m} \lambda_i} + (-1)^{m-k+1} \times \sum_{i=k, m} \left[\frac{1}{\lambda_i} \exp(-\lambda_i t) \prod_{j=k, m; j \neq i} \left(\frac{1}{\lambda_i - \lambda_j} \right) \right] \quad (25)$$

**Fig. 5.** Flow chart of major processes in the Activity code.

$$g(t; k, m, \vec{\lambda}) = \frac{1}{\prod_{i=k,m} \lambda_i} t + \sum_{i=k,m} \left[\prod_{j=k,m; j \neq i} \lambda_j \right] \\ + (-1)^{m-k+1} \sum_{i=k,m} \left[\frac{1}{\lambda_i^2} \exp(-\lambda_i t) \prod_{j=k,m; j \neq i} \left(\frac{1}{\lambda_i - \lambda_j} \right) \right]. \quad (26)$$

1.10. Preference: analytic over numeric

The numeric solution only requires the equation to be solved in the s-domain; the Gaver–Stehfest algorithm performs the inversion. It is worth the extra effort to derive and implement an analytic solution, as the numeric is only an approximation. Examples of the pitfalls of the numeric solution are that it can give negative amounts of an isotope and the difference between the numeric and analytic calculated amounts can become quite large when the isotope decays away to a very small value. Fig. 4 shows the predicted decay of a sample of Po-218 irradiated for 1000 s, and sampled until 10,000 s. In the region between 4000 s and 9000 s the amount from the numeric calculation drops below zero, whereas the analytic calculation remains above zero, as would be expected.

2. Computational methods

The Activity program has been developed in Fortran and takes advantage of MPI (Message Parsing Interface) to speed up calculation times by allowing the use of multiple processes in parallel. It has a self contained maths library, although this could be improved in the future by using optimized maths libraries for certain functions (e.g. matrix operations).

The code was developed on a Debian based distribution of Linux, but it should be supported on other variants of Linux and Unix, and does not require any specialist hardware.

The user is required to prepare an input file that contains the instructions required to perform a calculation. In addition to the input file, the user must provide an EXYZ ion trajectory file output by SRIM. Activity will read in the user input file, and the SRIM and data files listed within, before performing the calculation. Fig. 5 shows a flowchart of the major steps the code performs.

There are various settings in the user input file, but the main ones relating to the simulated experiment are:

- Element composition of target (percentage by mass).
- Beam flux (current), energy, duration and area on target.
- Activity measurement time (end of the “experiment”).
- Material density.
- Target thickness.

Several data files are generated by Activity and, if the user has matplotlib [7], charts will be created too. The most relevant to the user are:

- gammaLines.dat – tally into discrete bins of predicted gamma counts.
- ionTraj.dat – the averaged ion trajectory used in the calculation.
- isotopeActivityFileG.dat – a large data file detailing the activity of every predicted radioactive isotope in the target at user specified times following irradiation.

The charts include:

- activityTop5.png – activity of the top 5 active isotopes as a function of time after irradiation starts.
- gammaLines.png – predicted gamma spectrum expected at the “experiment end time”.

The Activity code uses the equations derived above to calculate the amount and activity of each isotope in the calculation. One problem with the original Bateman equation that also exists in the set of modified Bateman equations is that two different isotopes with the same decay constant will cause a singularity and a halt in the calculation. The activity code loops through all the decay constants in use before it attempts to run the calculation. If any isotope decay constants match they are varied by a small amount relative to the decay constant. It repeats this process until all decay constants are unique before proceeding.

3. Approximations

The accuracy of the Activity code is dependent on the input files provided by the user and the method used to calculate the reaction rates and resulting activity. The TENDL proton database consists of experimental measured cross sections as well as values calculated using the optical model potential. Using the latest database is recommended.

SRIM uses the binary collision approximation to simulate ion transport. It is a well tested code that has been used for many years. One limitation is that the structure of the material is not taken into account. This would have an impact on a user of the Activity program if they were trying to calculate, for example, whether a FCC (face centered cubic) steel would be irradiated differently when compared to a BCC (body centered cubic) steel. The Activity code would determine the activity of the steel as a function of the ion current, ion type and the density, thickness and composition of the steel, not its structure.

This version of the Activity code averages the path of all the SRIM simulated ions, rather than treating each ion differently. This may or may not have an impact on the results. If a new version of the code is developed there would be an option to calculate reaction rates for each individual simulated ion, and a comparison could then be made to the calculations using the averaged path of a set of ions.

The final approximation would be to use the numeric solution to the activity equations, although the analytic solution is forced within the code unless it returns a failed result.

4. Results

A target of high purity Iron was irradiated with 36 MeV protons by the University of Birmingham Scanditronix MC-40 Cyclotron. The target was 0.5 mm thick and was irradiated at a current of 0.5 μ A for 300 s, irradiating approximately 0.25 g of Iron. A high purity Germanium detector was used to measure the gamma peaks three days after irradiation.

The peak that dominated the readings was the 931 keV Cobalt 55 line. After calibrating the detector and adjusting the readings, this peak was measured at 44,300 Bq+/-2000 Bq. The activity of this peak as predicted by the Activity code was 44,565 Bq.

Table 1

Gamma peaks predicted and measured for a 13 MeV ion irradiated sample of Mo.

Gamma energy (keV)	Predicted activity (Bq)	Experimental activity (Bq)
766	4.45E6	5.11E5+/-2.5E4
778	6.14E6	1.36E6+/-6.8E4
812	5.04E6	1.15E6+/-5.8E4
850	6.00E6	1.39E6+/-7.0E4
126	9.33E5	2.10E5+/-1.1E4

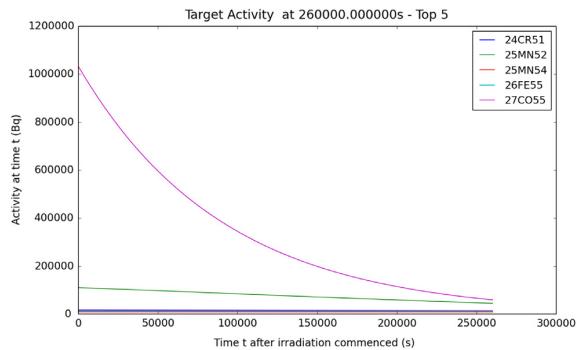


Fig. 6. Sample Activity code output chart for the top five most active isotopes for Iron irradiated by 36 MeV protons.

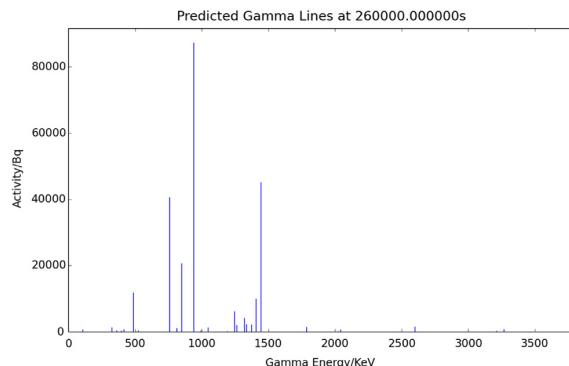


Fig. 7. Sample Activity code output chart for the expected gamma lines to be measured for Iron irradiated by 36 MeV protons.

A target of high purity Molybdenum was irradiated with 13 MeV protons by the University of Birmingham Scanditronix MC-40 Cyclotron. The target was 0.5 mm thick and was irradiated at a current of 5 μ A for 1500 s, irradiating approximately 0.3 g of Molybdenum. A high purity Germanium detector was used to measure the gamma peaks three days after irradiation.

Five peaks were of interest, and these are listed in Table 1.

There was the possibility of the high purity Germanium detector introducing errors due to detector dead time, where by a source that is too active floods the device with gammas. The samples were safe to handle, and did not appear to flood the detector in any way, but there was still the possibility of counts being missed due to dead time. The probabilistic nature of radioactive decay also introduced inherent errors to the experimental activity measurements (see Figs. 6–9).

5. Conclusions

The Activity program is an easy to use Fortran compiled executable that can be built and run, for free, on a Linux computer. It

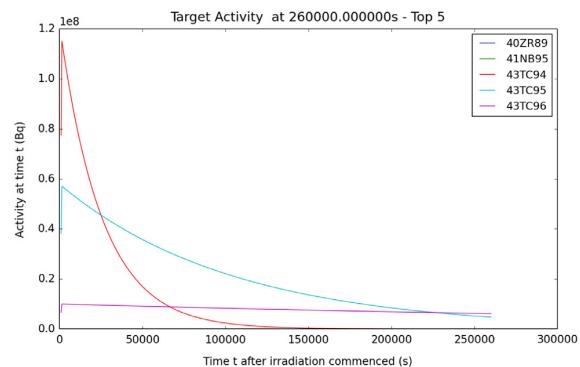


Fig. 8. Sample Activity code output chart for the top five most active isotopes for Molybdenum irradiated by 13 MeV protons.

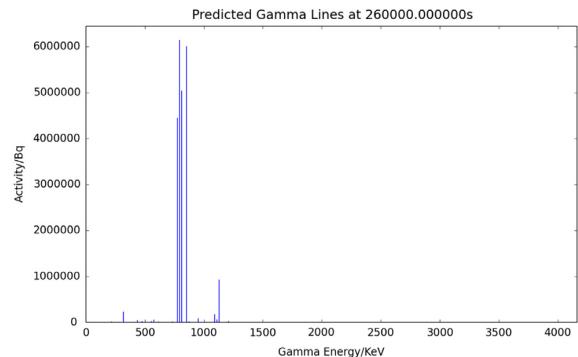


Fig. 9. Sample Activity code output chart for the expected gamma lines to be measured for Molybdenum irradiated by 13 MeV protons.

takes advantage of multi-core processors, and typical calculations take from seconds to minutes.

Using the SRIM code and TENDL database, the code has been used to predict the activity of Iron and Molybdenum targets that have been irradiated with a proton beam. The prediction of the 931 (keV) Cobalt 55 gamma activity of irradiated Iron was very close to the measured value. The five predicted gamma activities for ion irradiated Molybdenum were up to a factor of 5–10 away from the measured activities.

There are a number of improvements that would be considered in a future version of the Activity code. These improvements would include the following:

- A more readable output with less information printed, as too much may confuse the user.
- A file containing the top five radioactive isotopes with their top five gamma lines.
- Individual ion trajectories used to calculate the reaction rates, rather than the average path.
- Experimental ion activation data for a wider range of elements to test the Activity code against.
- Expand to include deuterons (this would require the TENDL deuteron cross section database).

Acknowledgments

We would like to thank and acknowledge the input and advice of the following:

- Dr Chris Cooper and John Hewett for irradiation activation data points.

- The University of Birmingham for providing the funding for this project.

References

- [1] A.J. Koning, D. Rochman, Nucl. Data Sheets 113 (2012).
- [2] A.A. Sonzogni, Nuclear Data Services. www-nds.iaea.org/ndspub/download-endf/ENDF-B-VII.0/decay-index.htm (visited on 08/14/2015).
- [3] M. Herman, M.B. Chadwick, Nucl. Data Sheets 107 (2006) 2931.
- [4] J.P. Biersack, J.F. Ziegler, M.D. Ziegler, Nuclear Instrum. Methods Phys. Res. B 268 (2010) 1818–1823.
- [5] P. Blaise, M. Coste, A. Courcelle, T.D. Huynh, C. Jouanne, P. Leconte, O. Litaize, S. Mengelle, G. Nogure, J.-M. Ruggiri, O. Srot, J. Tommasi, C. Vaglio, J.-F. Vidal, A. Santamarina, D. Bernard, The JEFF-3.1.1 Nuclear Data Library. 2009. ISBN: 978-92-64-99074-6.
- [6] H. Stehfest, Commun. ACM 13 (1970) 47–49.
- [7] J.D. Hunter, Comput. Sci. Eng. 9 (2007) 90–95.

C.1.2 Accompanying Manual

The accompanying manual for the Activity code is included in the following pages.

UNIVERSITY OF BIRMINGHAM



Department of Metallurgy & Materials

Activity Manual

Contents

1 Background	4
1.1 Ion Irradiation	4
1.1.1 Ion Irradiation at the University of Birmingham	4
1.1.2 Simulating Ion Irradiation with SRIM	4
1.1.3 Transmutation of Nuclei by Ion Irradiation	5
1.2 Decay and Activity Equations	6
1.2.1 Radioactive Decay	6
1.2.2 Bateman Equation for Radioactive Decay	6
1.2.3 Laplace Transform	7
1.2.4 Constructing the Differential Equations	7
1.2.5 Numerical Inversion of the Laplace Transform	8
1.2.6 Analytic Solution by Partial Fraction Expansion	9
1.2.7 Preference: Analytic over Numeric	11
1.3 Computational Methods	12
1.3.1 Activity Code	12
1.3.2 Approximations	13
1.3.3 Results	13
2 Installation and Using the Activity Code	17
2.1 Getting Started	17
2.1.1 Prerequisites	17
2.2 Installing Activity	17
2.2.1 Download Source Code	17
2.2.2 Compile Source Code	18
2.3 Input File	18
2.4 Acknowledgements	21
Appendices	22

A Example Input File	23
A.1 Iron 36MeV Proton Beam	23
B Fortran 90 Code	25
B.1 Fortran 90 Implementation of Analytic Method	25

List of Figures

1.1	Proton energy loss in Fe simulated with SRIM [2]	5
1.2	An example decay chain from an unstable parent isotope, through unstable daughter isotopes ending with a stable daughter isotope.	6
1.3	An example of several decay chains including branching factors and possible external source terms for each isotope on each chain.	7
1.4	Decay of Po-218: Analytice and Gaver-Stehfest Calculations [5]	11
1.5	Flow chart of major processes in the Activity code	12
1.6	Sample Activity code output chart for the top five most active isotopes for Iron irradiated by 36MeV protons.	14
1.7	Sample Activity code output chart for the expected gamma lines to be measured for Iron irradiated by 36MeV protons.	15
1.8	Sample Activity code output chart for the top five most active isotopes for Molybdenum irradiated by 13MeV protons.	15
1.9	Sample Activity code output chart for the expected gamma lines to be measured for Molybdenum irradiated by 13MeV protons.	16

Chapter 1

Background

1.1 Ion Irradiation

A computer program, Activity, was developed to predict the activity and gamma lines of materials irradiated with an ion beam. It uses the TENDL[1] proton reaction cross section database, the Stopping and Range of Ions in Matter (SRIM)[2] code, a Nuclear Data Services (NDS) radioactive decay database [3] and an ENDF gamma decay database [4]. An extended version of Bateman's equation is used to calculate the activity at time t , and this equation is solved analytically, with the option to also solve by numeric inverse Laplace transform as a failsafe. The program outputs the expected activity and gamma lines of the activated material.

1.1.1 Ion Irradiation at the University of Birmingham

The Scanditronix MC-40 Cyclotron is used at the University of Birmingham to create a beam of protons or other light ions. The energies of these ions are typically between 10 MeV and 60 MeV with beam currents ranging up to 50 microamps (3.1×10^{14} protons per second). Target materials are irradiated by this cyclotron for a number of reasons, including purposely creating radioactive isotopes for the nearby Queen Elizabeth Hospital, investigating ion irradiation damage and emulating neutron irradiation.

The Cyclotron is usually used to create radioactive isotopes for medical use, but an additional beam line has been devoted to material science investigations into radiation damage. While the creation of radioactive isotopes is desired in some cases, material being tested for radiation damage should preferably have low levels of radioactivity.

It is expensive to arrange the irradiation of target materials by high energy neutrons sources, whereas it is relatively inexpensive to irradiate using an ion beam on the MC-40 Cyclotron. The energies can be controlled, and a set dose at a single energy, or a range of energies, can be precisely deposited in the target material.

The Activity code discussed here was developed to calculate the activity of a target material irradiated by a proton beam. It has been developed in Fortran and uses data from the TENDL-2013 proton cross section database, SRIM ion transport code and NDS radioactive decay database.

1.1.2 Simulating Ion Irradiation with SRIM

A package of ion transport codes, SRIM, is freely available to download and use to investigate the transport of ions through matter. SRIM uses the binary collision approximation (BCA) to simulate the passage of ions in a material. It is an approximate method, and one key restriction is that it does not take into account the structure of the material, and this approximation is therefore also imposed on the Activity code.

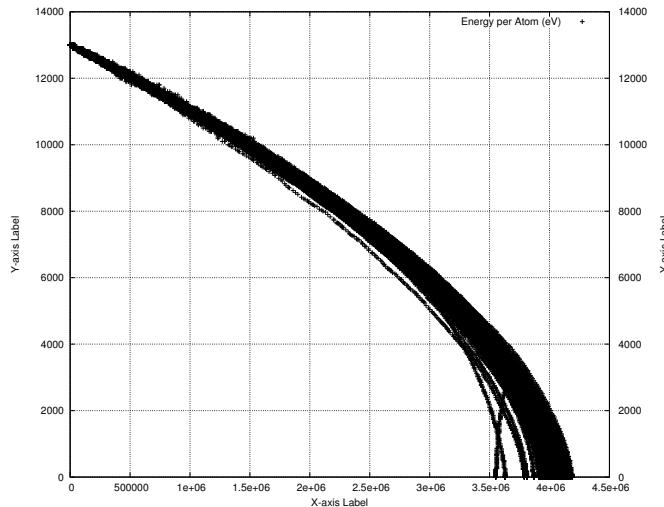


Figure 1.1: Proton energy loss in Fe simulated with SRIM [2]

One file that SRIM creates is of importance to the Activity code, and that is the trajectory file that contains the energy and x,y,z co-ordinate data points for simulated ions moving through matter. Figure 1.1 shows the trajectory of protons passing through an Iron target, and it is this set of data points (together with the cross section database) that the Activity code uses to calculate the reaction rates for the transmutation of nuclei in the target. At higher energies, the ions slow as they lose energy due to electronic stopping, but as the ion energy drops the mechanism of loss through nuclear collisions becomes important. The spreading of ion depths at lower energies is a result of the higher momentum transfer during nuclear collisions, as can be seen in Figure 1.1.

1.1.3 Transmutation of Nuclei by Ion Irradiation

Considering a simplified nuclear potential well, energetic protons approaching a nucleus may overcome the coulomb potential barrier. They are captured by the nucleus and held within the potential well by the strong nuclear force. This process may leave the nucleus in an excited and unstable state, depending on the input energy of the proton and configuration of nucleons. The process is probabilistic, and the average chance of a reaction (the microscopic cross section) may be measured as a function of the projectile, projectile energy and target, either experimentally or by optical model potential calculations. The reaction rate is calculated from the microscopic cross section using the following equation:

$$R = \frac{J}{e} n_t \sigma \cdot 10^{28} \delta t \quad (1.1)$$

- R Reaction Rate (reactions per second)
- J Beam current (A)
- n_t Number density of target (atoms per cubic metre)
- σ Microscopic reaction cross section (barns)
- e Elementary charge (1.602177E-19C)
- δT Target thickness (m)

1.2 Decay and Activity Equations

1.2.1 Radioactive Decay

Radioactive decay is the random change in nucleons or energy state of an unstable nucleus. It is impossible to predict when a nucleus will decay, but the decay of a collection of nuclei is statistical in nature. The radioactivity and number of unstable nuclei at time t can be predicted using the decay constant λ for the radioactive isotope. This constant is defined as follows:

$$\lambda = -\frac{N'(t)}{N(t)} \quad (1.2)$$

The number of radioactive nuclei $N(t)$ at time t is given by the following equation, where $N(0)$ is the starting number of nuclei:

$$N(t) = N(0) \exp(-t\lambda) \quad (1.3)$$

The activity $A(t)$ of the radioactive nuclei is predicted at time t by using the following equations, where $N'(t)$ is the change in amount of nuclei with respect to time:

$$A(t) = -N'(t) = \lambda N(t) \quad (1.4)$$

$$A(t) = \lambda N(0) \exp(-t\lambda) \quad (1.5)$$

1.2.2 Bateman Equation for Radioactive Decay

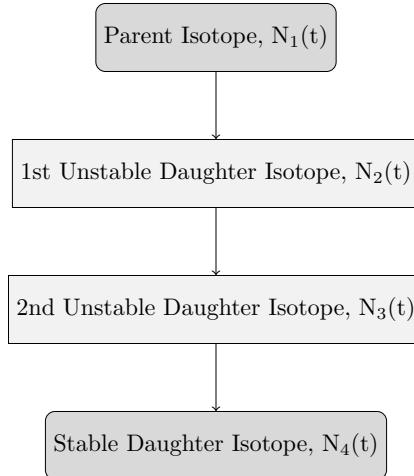


Figure 1.2: An example decay chain from an unstable parent isotope, through unstable daughter isotopes ending with a stable daughter isotope.

The English mathematician Harry Bateman derived an equation to calculate the amount of each isotope in a decay chain, illustrated in Figure 1.2, at time t .

$$N_n(t) = \sum_{i=1}^{i=n} \left(\left(\prod_{j=i}^{j=n-1} \lambda_{(ij+1)} \right) \sum_{j=i}^{j=n} \left(\frac{N_{i0} \exp(-\lambda_j t)}{\prod_{p=i, p \neq j}^p (\lambda_p - \lambda_j)} \right) \right) \quad (1.6)$$

When a radioactive isotope decays, there may be more than one mode of decay, and this leads to branching factors. Pb-214 only decays via beta decay to Bi-214, giving a branching factor of 1.0, whereas Bi-214 has a 99.979% chance of decaying to Po-214 by beta decay and a 0.021% of emitting an alpha particle and decaying to Tl-210 (branching factors of 0.99979 and 0.00021 respectively) [5].

When a target material is irradiated, there is a source term for transmuted nuclei due to the irradiation. The daughter isotopes of these transmuted isotopes will also be affected by the irradiation and will transmute further, giving a source term for each daughter isotope as a result of the irradiation. Sources for each isotope in the decay chain, and branching factors between a parent isotope and its daughter isotope/s must be accounted for.

The Bateman equation was derived using Laplace transforms, and this same method has been used to develop a modified equation that incorporates branching factors and production rates for each isotope in the decay chain, as illustrated by Figure 1.3.

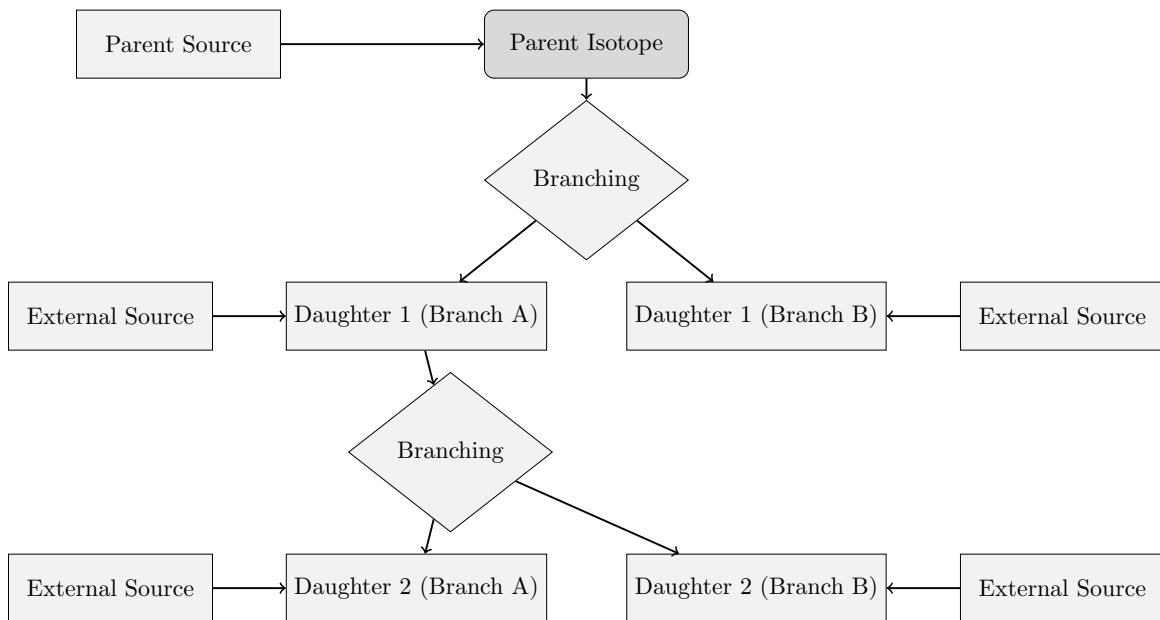


Figure 1.3: An example of several decay chains including branching factors and possible external source terms for each isotope on each chain.

1.2.3 Laplace Transform

Laplace Transforms are a useful mathematical tool, and allow ordinary differential equations to be solved by simple algebraic manipulation in the s domain. Bateman took advantage of Laplace Transforms in deriving his equation, and this is the method that has been taken here as well.

$$F(s) = \int_0^{\infty} f(t) \exp(-st) dt \quad (1.7)$$

1.2.4 Constructing the Differential Equations

The first step is to set up differential equations for the parent isotope, unstable daughter isotopes and stable daughter isotope. The parent isotope has a source term, due to production, and a loss term, due to decay. The unstable daughter isotopes have two source terms, from the production by irradiation induced transmutation

and the decay of preceding isotopes in the decay chain, and a loss term, due to decay. Finally, the stable daughter that finalizes the decay chain has two source terms (the same as the unstable daughters) but no loss term.

The variables (and vectors) used in these equations are defined as follows:

- $\vec{\lambda}$ vector containing isotope decay constants λ_i
- \vec{b} vector containing isotope to isotope branching factors b_i
- \vec{w} vector containing isotope production rates w_i
- t time at which activity/amount of isotope is measured
- $N_i(0)$ starting amount of the i th isotope
- $N_i(t)$ amount of the i th isotope at time t
- $N'_i(t)$ change in amount of the i th isotope, with respect to time, at time t

The differential equations for the parent isotope (first isotope), unstable daughter isotopes (i th isotopes) and stable, final, daughter isotope (z th isotope) in the time domain are as follows:

$$N'_1(t) = \omega_1 - \lambda_1 N_1(t) \quad (1.8)$$

$$N'_i(t) = \omega_i + b_{i-1} \lambda_{i-1} N_{i-1}(t) - \lambda_i N_i(t) \quad (1.9)$$

$$N'_z(t) = \omega_z + b_{z-1} \lambda_{z-1} N_{z-1}(t) \quad (1.10)$$

Applying the Laplace Transform to these three differential equations allows them to be manipulated and solved algebraically in the s-domain.

$$N_1(s) = \frac{1}{s + \lambda_1} N_1(0) + \frac{1}{s(s + \lambda_1)} \omega_1 \quad (1.11)$$

$$N_i(s) = \frac{1}{s(s + \lambda_i)} (\omega_i) + \frac{1}{s + \lambda_i} (b_{i-1} \lambda_{i-1} N_{i-1}(s)) + \frac{1}{s + \lambda_i} N_i(0) \quad (1.12)$$

$$N_z(s) = \frac{1}{s^2} \omega_z + \frac{1}{s} b_{z-1} \lambda_{z-1} N_{z-1}(s) + \frac{1}{s} N_z(0) \quad (1.13)$$

1.2.5 Numerical Inversion of the Laplace Transform

The Gaver-Stehfest[6] algorithm was developed in the 1960s and 1970s and is a method of calculating the inverse of a Laplace Transform in the real number domain. It is an easy to implement and reasonably accurate method, although it is an approximation to the real value. A comparison between an analytic and numeric inversion for the unstable isotope Po-218 is discussed at the end of this section.

$$f(t) \approx f_n(t) = \frac{\ln(2)}{t} \sum_{k=1}^{2n} a_k(n) F(s) \text{ where } n \geq 1, t > 0 \quad (1.14)$$

$$s = \frac{k \ln(2)}{t} \quad (1.15)$$

$$a_k(n) = \frac{(-1)^{(n+k)}}{n!} \sum_{j=\text{Floor}(\frac{k+1}{2})} j^{n+1} \binom{n}{j} \binom{2j}{j} \binom{j}{k-j} \quad (1.16)$$

The equation for the i th isotope may be calculated by recursively calculating the equations by numeric inversion, starting from the first (parent isotope) and inserting the result into each subsequent recursion until the i th isotope is reached (changing the equations appropriately for the parent, unstable daughter and stable daughter isotopes).

1.2.6 Analytic Solution by Partial Fraction Expansion

The equation for the i th isotope in the s domain can be written in full by substituting the preceding equation until the parent isotope is reached, and this full equation may be rearranged with the production amount of each isotope and starting amount of each isotope in individual terms. Each of these terms is multiplied by a fraction that can be expanded, using partial fractions, and inverted analytically.

This is illustrated with an example unstable isotope, fourth in the decay chain (including the parent isotope):

$$\begin{aligned} N_4(s) &= \frac{1}{(s + \lambda_1)(s + \lambda_2)(s + \lambda_3)(s + \lambda_4)} b_2 b_3 b_4 \lambda_1 \lambda_2 \lambda_3 N_1(0) \\ &\quad + \frac{1}{(s + \lambda_2)(s + \lambda_3)(s + \lambda_4)} b_3 b_4 \lambda_2 \lambda_3 N_2(0) \\ &\quad + \frac{1}{(s + \lambda_3)(s + \lambda_4)} b_4 \lambda_3 N_3(0) \\ &\quad + \frac{1}{(s + \lambda_4)} N_4(0) \\ &\quad + \frac{1}{s(s + \lambda_1)(s + \lambda_2)(s + \lambda_3)(s + \lambda_4)} b_2 b_3 b_4 \lambda_1 \lambda_2 \lambda_3 \omega_1 \\ &\quad + \frac{1}{s(s + \lambda_2)(s + \lambda_3)(s + \lambda_4)} b_3 b_4 \lambda_2 \lambda_3 \omega_2 \\ &\quad + \frac{1}{s(s + \lambda_3)(s + \lambda_4)} b_4 \lambda_3 \omega_3 \\ &\quad + \frac{1}{s(s + \lambda_4)} \omega_4 \end{aligned} \quad (1.17)$$

An example stable isotope, fourth (last) in the decay chain (including the parent isotope):

$$\begin{aligned}
N_4(s) = & \frac{1}{s(s + \lambda_1)(s + \lambda_2)(s + \lambda_3)} b_2 b_3 b_4 \lambda_1 \lambda_2 \lambda_3 N_1(0) \\
& + \frac{1}{s(s + \lambda_2)(s + \lambda_3)} b_3 b_4 \lambda_2 \lambda_3 N_2(0) \\
& + \frac{1}{s(s + \lambda_3)} b_4 \lambda_3 N_3(0) \\
& + N_4(0) \\
& + \frac{1}{s^2(s + \lambda_1)(s + \lambda_2)(s + \lambda_3)} b_2 b_3 b_4 \lambda_1 \lambda_2 \lambda_3 \omega_1 \\
& + \frac{1}{s^2(s + \lambda_2)(s + \lambda_3)} b_3 b_4 \lambda_2 \lambda_3 \omega_2 \\
& + \frac{1}{s^2(s + \lambda_3)} b_4 \lambda_3 \omega_3 \\
& + \frac{1}{s^2} \omega_4
\end{aligned} \tag{1.18}$$

By using partial fraction expansion and standard Laplace Transforms, the set of equations below is used to calculate the amount of the mth isotope in the decay chain, providing the mth isotope is unstable.

$$N_m(t; \vec{\lambda}, \vec{b}, \vec{w}) = \sum_{k=1, m} r(k; \vec{\lambda}, \vec{b}) [f(t; k, m, \vec{\lambda}) N_k(0) + g(t; k, m, \vec{\lambda}) w_k] \tag{1.19}$$

$$r(k, m, \vec{\lambda}) = \begin{cases} \prod_{i=k, m-1} (b_{i+1} \lambda_i), & \text{if } k < m \\ 1, & \text{if } k = m \end{cases} \tag{1.20}$$

$$f(t; k, m, \vec{\lambda}) = (-1)^{m-k} \sum_{i=k, m} \left[\exp(-\lambda_i t) \prod_{j=k, m; j \neq i} \left(\frac{1}{\lambda_i - \lambda_j} \right) \right] \tag{1.21}$$

$$g(t; k, m, \vec{\lambda}) = \frac{1}{\prod_{i=k, m} \lambda_i} + (-1)^{m-k+1} \sum_{i=k, m} \left[\frac{1}{\lambda_i} \exp(-\lambda_i t) \prod_{j=k, m; j \neq i} \left(\frac{1}{\lambda_i - \lambda_j} \right) \right] \tag{1.22}$$

The set of equations below is used to calculate the amount of the mth isotope in the decay chain, where the mth isotope is stable.

$$N_m(t; \vec{\lambda}, \vec{b}, \vec{w}) = N_m + w_m t + \sum_{k=1, m-1} r(k; \vec{\lambda}, \vec{b}) [f(t; k, m-1, \vec{\lambda}) N_k(0) + g(t; k, m, \vec{\lambda}) w_k] \tag{1.23}$$

$$r(k, m, \vec{\lambda}) = \begin{cases} \prod_{i=k, m-1} (b_{i+1} \lambda_i), & \text{if } k < m \\ 1, & \text{if } k = m \end{cases} \tag{1.24}$$

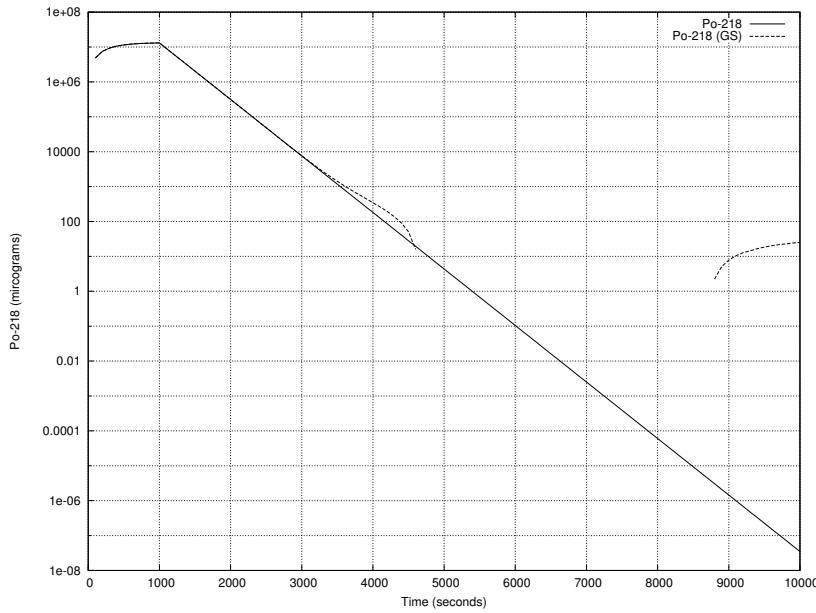


Figure 1.4: Decay of Po-218: Analytic and Gaver-Stehfest Calculations [5]

$$f(t; k, m, \vec{\lambda}) = \frac{1}{\prod_{i=k,m} \lambda_i} + (-1)^{m-k+1} \sum_{i=k,m} \left[\frac{1}{\lambda_i} \exp(-\lambda_i t) \prod_{j=k,m; j \neq i} \left(\frac{1}{\lambda_i - \lambda_j} \right) \right] \quad (1.25)$$

$$g(t; k, m, \vec{\lambda}) = \frac{1}{\prod_{i=k,m} \lambda_i} t + \frac{\sum_{i=k,m} \left[\prod_{j=k,m; j \neq i} \lambda_j \right]}{\prod_{i=k,m} \lambda_i^2} + (-1)^{m-k+1} \sum_{i=k,m} \left[\frac{1}{\lambda_i^2} \exp(-\lambda_i t) \prod_{j=k,m; j \neq i} \left(\frac{1}{\lambda_i - \lambda_j} \right) \right] \quad (1.26)$$

1.2.7 Preference: Analytic over Numeric

The numeric solution only requires the equation to be solved in the s-domain; the Gaver-Stehfest algorithm performs the inversion. It is worth the extra effort to derive and implement an analytic solution, as the numeric is only an approximation. Examples of the pitfalls of the numeric solution are that it can give negative amounts of an isotope and the difference between the numeric and analytic calculated amounts can become quite large when the isotope decays away to a very small value. Figure 1.4 shows the predicted decay of a sample of Po-218 irradiated for 1,000s, and sampled until 10,000s. In the region between 4,000s and 9,000s the amount from the numeric calculation drops below zero, whereas the analytic calculation remains above zero, as would be expected.

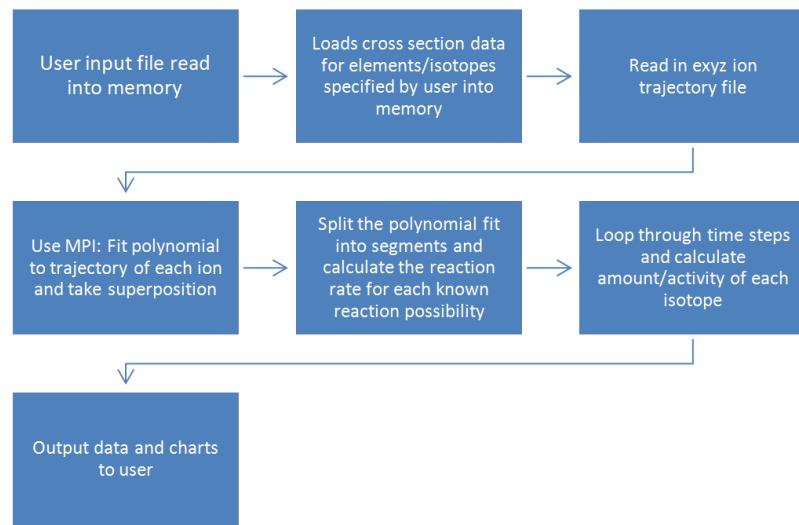


Figure 1.5: Flow chart of major processes in the Activity code

1.3 Computational Methods

1.3.1 Activity Code

The Activity program has been developed in Fortran and takes advantage of MPI (Message Parsing Interface) to speed up calculation times by allowing the use of multiple processes in parallel. It has a self contained maths library, although this could be improved in the future by using optimised maths libraries for certain functions (e.g. matrix operations).

The code was developed on a Debian version of Linux, but it should be supported on other variants of Linux and Unix, and does not require any specialist hardware.

The user is required to prepare an input file that contains the instructions required to perform a calculation. In addition to the input file, the user must provide an EXYZ ion trajectory file output by SRIM. Activity will read in the user input file, and the SRIM and data files listed within, before performing the calculation. Figure 1.5 shows a flowchart of the major steps the code performs.

There are various settings in the user input file, but the main ones relating to the simulated experiment are:

- element composition of target (percentage by mass)
- beam flux (current), energy, duration and area on target
- activity measurement time (end of the “experiment”)
- material density
- target thickness

Several data files are generated by Activity and, if the user has matplotlib [7], charts will be created too. The most relevant to the user are:

- gammaLines.dat - tally into discrete bins of predicted gamma counts

- ionTraj.dat - the averaged ion trajectory used in the calculation
- isotopeActivityFileG.dat - a large data file detailing the activity of every predicted radioactive isotope in the target at user specified times following irradiation

The charts include:

- activityTop5.png - activity of the top 5 active isotopes as a function of time after irradiation starts
- gammaLines.png - predicted gamma spectrum expected at the “experiment end time”

The Activity code uses the equations derived above to calculate the amount and activity of each isotope in the calculation. One problem with the original Bateman equation that also exists in the set of modified Bateman equations is that two different isotopes with the same decay constant will cause a singularity and a halt in the calculation. The activity code loops through all the decay constants in use before it attempts to run the calculation. If any isotope decay constants match they are varied by a small amount relative to the decay constant. It repeats this process until all decay constants are unique before proceeding.

1.3.2 Approximations

The accuracy of the Activity code is dependant on the input files provided by the user and the method used to calculate the reaction rates and resulting activity. The TENDL proton database consists of experimental measured cross sections as well as values calculated using the optical model potential. Using the latest database is recommended.

SRIM uses the binary collision approximation to simulate ion transport. It is a well tested code that has been used for many years. One limitation is that the structure of the material is not taken into account. This would have an impact on a user of the Activity program if they were trying to calculate, for example, whether a FCC (face centered cubic) steel would be irradiated differently when compared to a BCC (body centered cubic) steel. The Activity code would determine the activity of the steel as a function of the ion current, ion type and the density, thickness and composition of the steel, not its structure.

This version of the Activity code averages the path of all the SRIM simulated ions, rather than treating each ion differently. This may or may not have an impact on the results. If a new version of the code is developed there would be an option to calculate reaction rates for each individual simulated ion, and a comparison could then be made to the calculations using the averaged path of a set of ions.

The final approximation would be to use the numeric solution to the activity equations, although the analytic solution is forced within the code unless it returns a failed result.

1.3.3 Results

A target of high purity Iron was irradiated with 36 MeV protons by the University of Birmingham Scanditronix MC-40 Cyclotron. The target was 0.5mm thick and was irradiated at a current of 0.5 micro Amps for 300 seconds, irradiating approximately 0.25g of Iron. A high purity Germanium detector was used to measure the gamma peaks three days after irradiation.

The peak that dominated the readings was the 931 keV Cobalt 55 line. After calibrating the detector and adjusting the readings, this peak was measured at 44,300Bq+/-2,000Bq. The activity of this peak as predicted by the Activity code was 44,565Bq.

A target of high purity Molybdenum was irradiated with 13 MeV protons by the University of Birmingham Scanditronix MC-40 Cyclotron. The target was 0.5mm thick and was irradiated at a current of 5 micro Amps

Table 1.1: Gamma peaks predicted and measured for a 13 MeV ion irradiated sample of Mo

Gamma Energy (keV)	Predicted Activity (Bq)	Experimental Activity (Bq)
766	4.45E6	5.11E5 +/- 2.5E4
778	6.14E6	1.36E6 +/- 6.8E4
812	5.04E6	1.15E6 +/- 5.8E4
850	6.00E6	1.39E6 +/- 7.0E4
126	9.33E5	2.10E5 +/- 1.1E4

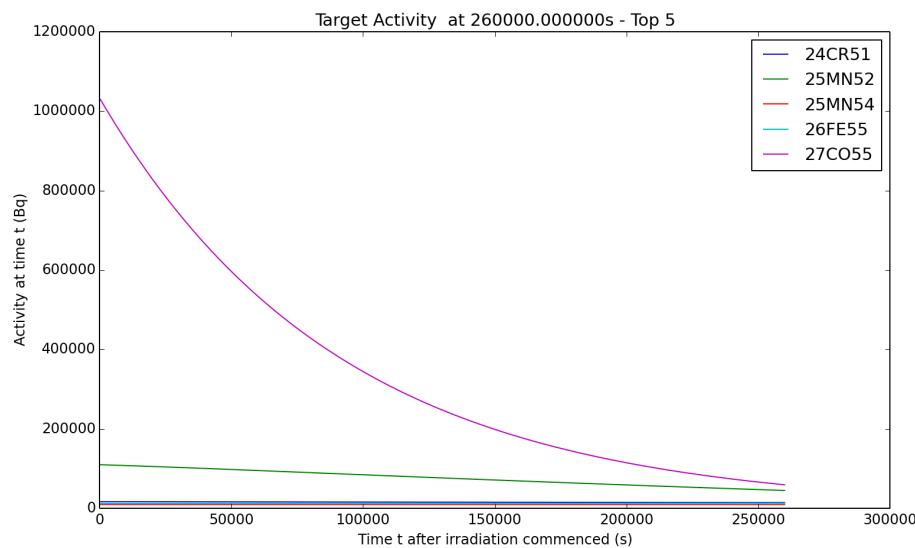


Figure 1.6: Sample Activity code output chart for the top five most active isotopes for Iron irradiated by 36MeV protons.

for 1500 seconds, irradiating approximately 0.3g of Molybdenum. A high purity Germanium detector was used to measure the gamma peaks three days after irradiation.

Five peaks were of interest, and these are listed in Table 1.1.

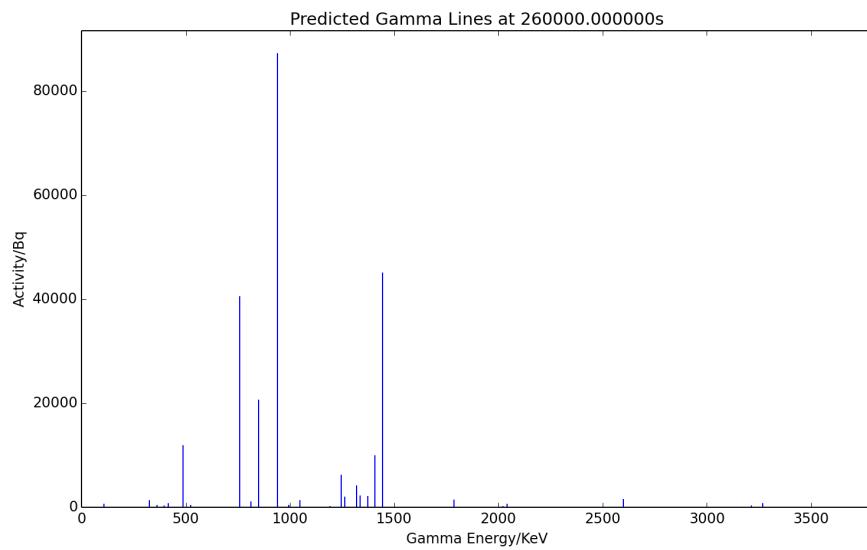


Figure 1.7: Sample Activity code output chart for the expected gamma lines to be measured for Iron irradiated by 36MeV protons.

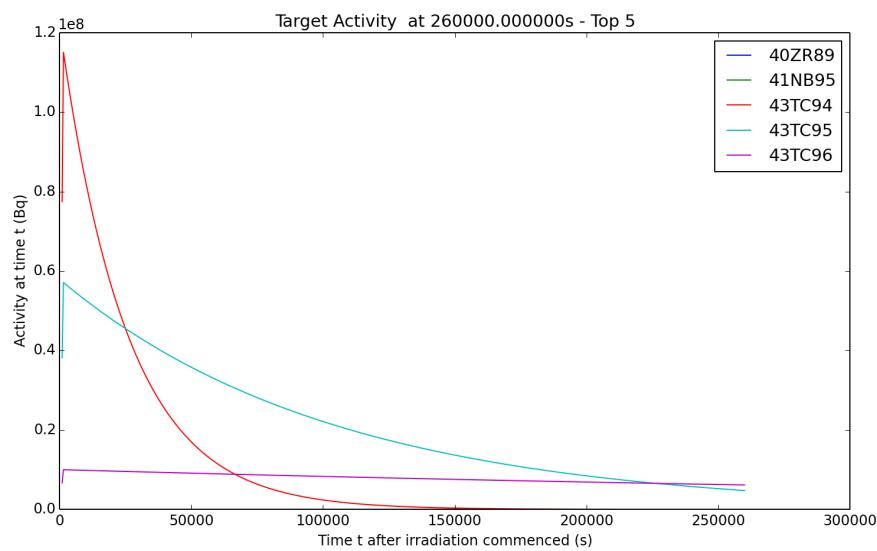


Figure 1.8: Sample Activity code output chart for the top five most active isotopes for Molybdenum irradiated by 13MeV protons.

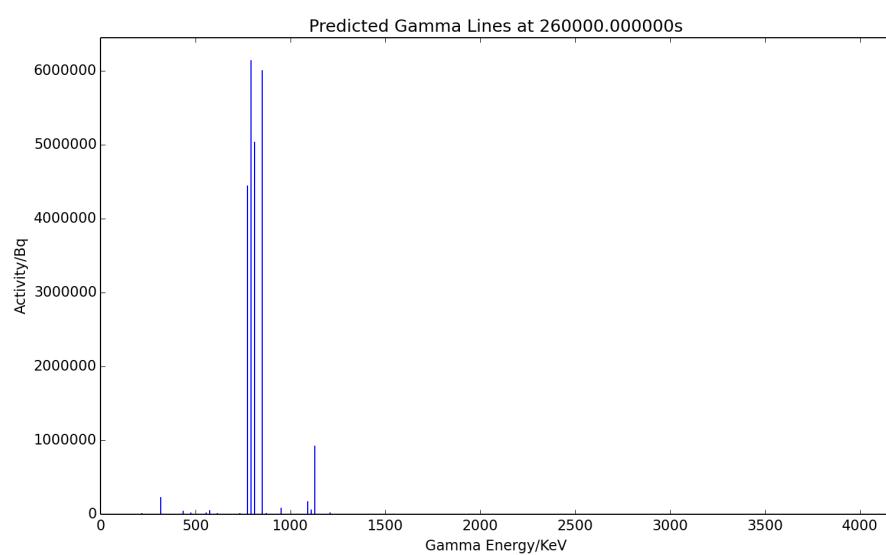


Figure 1.9: Sample Activity code output chart for the expected gamma lines to be measured for Molybdenum irradiated by 13MeV protons.

Chapter 2

Installation and Using the Activity Code

2.1 Getting Started

2.1.1 Prerequisites

This code was designed to run on Linux and has been developed and tested on the Debian based distribution Mint. It has not been tested on any other versions of Linux or Unix, nor has it been tested on cygwin.

The minimal requirements of the code are:

- 250MB per process + 200MB overhead
- 1GB Hard Drive space
- Ideally a modern multicore processor
- Fortran90
- OpenMPI

Optional:

- GNUPlot
- Python & MatPlotLib

2.2 Installing Activity

2.2.1 Download Source Code

The most recent source code is available for download from github. It may be downloaded and extracted using the terminal and the commands in listing 1.

Listing 2.1: bash

```

1 cd ~
2 mkdir -p ~/activity/tar
3 cd ~/activity/tar
4 wget https://github.com/BenPalmer1983/activity/raw/master/activity.tar.gz
5 tar -xzvf activity.tar.gz

```

2.2.2 Compile Source Code

Once the source files have been downloaded and extracted, run the install script then remove the downloaded files (if you need to free up space). The terminal commands are given in listing 2. Providing the prerequisites are available, the process will only take a few minutes (with a reasonable Internet connection).

Listing 2.2: bash

```

1 cd ~/activity/tar
2 ./install.sh #choose installation directory when prompted; default is ~/activity
3 rm -R ~/activity/tar

```

The activity code is now installed and the activity.x binary may be executed on the terminal. If the terminal fails to find the binary, the .bashrc or .bash_profile file must be edited, or a symlink created from the bin file to a valid and accessible bin directory.

2.3 Input File

The input file sets the target starting composition, the beam and simulated experiment settings as well as the paths to the required data files.

#elements

The elements keyword instructs the code to read in the composition of the target material. The elements and their percentage by weight are listed under the keyword.

Listing 2.3: Input File

```

1 #elements
2 Fe 72
3 Cr 18
4 Ni 8
5 Mg 2

```

#isotopes #decaymodes #gammaenergies #xsfiles

Four sets of data files are distributed with the code, and the paths to these files are defined underneath each keyword. N.B. the path for the xsfiles is to the directory that holds all the cross section data files, rather than each individual file.

Listing 2.4: Input File

```

1 #isotopes
2 "/home/ben/activity/data/isotopes.txt"
3 #decaymodes
4 "/home/ben/activity/data/decaymodes.txt"

```

```

5 #gammaenergies
6 "/home/ben/activity/data/gammaenergies.txt"
7 #xsfiles
8 "/home/ben/activity/data/xs"

```

#trajfile

The SRIM exyz file used for the calculation is pointed to under this keyword.

Listing 2.5: Input File

```

1 #trajfile
2 "/home/ben/activity/examples/Fe36MeV.exyz"

```

#polyfitorder #integrationgranularity

The code calculates the reaction rate of each projectile and target nucleus by first fitting a polynomial to the energy/depth data file. This polynomial gives an average path for the projectiles travelling through the target, allowing E(x) to be calculated quickly. The order of the polynomial is selected here, and a 5th order shold give a reasonable fit. The integration granularity keyword determines how many sections the polynomial is split up into. The energy, cross section for each section and the beam settings are used to calculate the reaction rate for each section, and these are all summed up to give the overall reaction rate.

Listing 2.6: Input File

```

1 #polyfitorder
2 5
3 #integrationgranularity
4 10

```

#beamflux #beamenergy #beamduration #beamarea

These keywords are self explanatory and control the beam settings used in the calculation.

Listing 2.7: Input File

```

1 #beamflux
2 0.5 uA
3 #beamenergy
4 36 MeV
5 #beamduration
6 300 s
7 #beamarea
8 100 mm2

```

#amtime #timestep

The time at which the activity of the sample is measured is set by the #amtime keyword. The Activity code calculates the activity of each radioactive isotope in the calculation from the time the beam starts until the activity measured time; and the intervals at which these calculations are made is determined by the #timestep keyword.

Listing 2.8: Input File

```

1 #amtime
2 260000 s

```

```
3 #timestep
4 1000 s
```

#projectile

The current version only supports protons as the projectile, therefore the only possible atomic and mass number combination for a projectile is 1 1.

Listing 2.9: Input File

```
1 #projectile
2 1 1
```

#targetthickness #materialdensity

Both the target thickness and density of the material it is made from are input with these keywords.

Listing 2.10: Input File

```
1 #targetthickness
2 0.5 mm
3 #materialdensity
4 8000 kgm3
```

#vpi

This keyword is only being used for a feature currently under testing, so the keyword can be ommited from the input file.

Listing 2.11: Input File

```
1 #vpi
2 60.2
```

#individualisotopeactivity #verboseterminal

If the #individualisotopeactivity keyword is followed by yes; additional data files are output containing the activity of each individual isotope in the calulcation. #verboseterminal followed by yeswill output more verbosely to the terminal.

Listing 2.12: Input File

```
1 #individualisotopeactivity
2 yes
3 #verboseterminal
4 yes
```

#targetdpa

This keyword is only being used for a feature currently under testing, so the keyword can be ommited from the input file.

Listing 2.13: Input File

```
1 #targetdpa
2 0.0
```

```
#gammachartresolution
```

The gamma chart is created by tallying the gamma values into bins, and this figure specifies the resolution(number of bins) used to create the output chart.

Listing 2.14: Input File

```
1 #gammachartresolution
2 200
```

2.4 Acknowledgements

We would like to thank and acknowledge the input and advice of the following:

- Dr Chris Cooper and John Hewett for irradiation activation data points.
- The University of Birmingham for providing the funding for this project.

Appendices

Appendix A

Example Input File

A.1 Iron 36MeV Proton Beam

Listing A.1: Fe36MeV.in

```

1 #elements
2 Fe 100
3 #isotopes
4 "/home/ben/activity/data/isotopes.txt"
5 #decaymodes
6 "/home/ben/activity/data/decaymodes.txt"
7 #gammaenergies
8 "/home/ben/activity/data/gammaenergies.txt"
9 #xsfiles
10 "/home/ben/activity/data/xs"
11 #trajfile
12 "/home/ben/activity/examples/Fe36MeV.exyz"
13 #polyfitorder
14 5
15 #integrationgranularity
16 10
17 #beamflux
18 0.5 uA
19 #beamenergy
20 36 MeV
21 #beadmduration
22 300 s
23 #beamarea
24 100 mm2
25 #amtime
26 260000 s
27 #timestep
28 1000 s
29 #projectile
30 1 1
31 #targetthickness
32 0.5 mm
33 #materialdensity
34 8000 kgm3

```

```
35 #vpi
36 60.2
37 #individual isotope activity
38 yes
39 #verbose terminal
40 yes
41 #targetdpa
42 0.0
43 #gammachart resolution
44 200
```

Appendix B

Fortran 90 Code

B.1 Fortran 90 Implementation of Analytic Method

Listing B.1: Fortran 90

```

1 Type :: decayChainObj
2   Real(kind=DoubleReal) :: time = 0.0D0
3   !Real(kind=DoubleReal) :: productionRate = 0.0D0
4   Integer(kind=StandardInteger) :: isotopes
5   Character(Len=16), Dimension(1:100) :: label
6   Real(kind=DoubleReal), Dimension(1:100) :: productionRate = 0.0D0
7   Real(kind=DoubleReal), Dimension(1:100) :: branchFactor = 1.0D0 ! from isotope parent
8   Real(kind=DoubleReal), Dimension(1:100) :: decayConstant = -1.0D0 ! negative for stable
9   Real(kind=DoubleReal), Dimension(1:100) :: halfLife = -1.0D0 ! negative for stable
10  Real(kind=DoubleReal), Dimension(1:100) :: amountStart = 0.0D0
11  Real(kind=DoubleReal), Dimension(1:100) :: amountEnd = 0.0D0
12 End Type
13
14 Subroutine CalcIsotopeChain(decayChain)
15 ! Uses inverse laplace transform to calculate isotope amounts at time t (after time = 0)
16 ! t time in seconds after t=0
17 ! w production rate of parent isotope
18 ! isotope chain data
19 Implicit None ! Force declaration of all variables
20 ! Vars In/Out
21 Type(decayChainObj) :: decayChain
22 ! Vars Private
23   Integer(kind=StandardInteger) :: i
24   Real(kind=DoubleReal) :: t, nEnd
25   Real(kind=DoubleReal), Dimension(1:100) :: W ! Production Rate
26   Real(kind=DoubleReal), Dimension(1:100) :: L ! Lambda
27   Real(kind=DoubleReal), Dimension(1:100) :: N ! Starting number of atoms
28   Real(kind=DoubleReal), Dimension(1:100) :: B ! Exp
29 ! Complete decay chain data
30   Call decayChainComplete(decayChain)
31 ! store input in shortned name arrays to make equations clearer
32   t = decayChain%time
33   Do i=1,decayChain%isotopes
34     W(i) = decayChain%productionRate(i)

```

```

35      L(i) = decayChain%decayConstant(i)
36      B(i) = decayChain%branchFactor(i)
37      N(i) = decayChain%amountStart(i)
38      End Do
39 ! Break infinities
40      Call DecayBreakInfinities(L,decayChain%isotopes)
41 ! Run analytic calculations
42 ! Loop through isotopes
43 ! Using L-1(1/(q+ps) = (1/p)*exp(-1*(q/p)*t) and partial fractions
44      Do i=1,decayChain%isotopes
45          nEnd = CalcIsotopeChainCalc(t,i,W,L,N,B)
46          decayChain%amountEnd(i) = nEnd
47      End Do
48 End Subroutine CalcIsotopeChain
49
50 Subroutine decayChainComplete(decayChain)
51 ! Completes the decay chain object
52 Implicit None ! Force declaration of all variables
53 ! Vars In/Out
54 Type(decayChainObj) :: decayChain
55 ! Vars Private
56 Integer(kind=StandardInteger) :: i, n
57 n = 0
58 Do i=1,100
59     n = n + 1
60     If(decayChain%decayConstant(i).eq.-1.0D0.and.decayChain%halfLife(i).gt.0.0D0)Then
61 ! complete decay constant from half life
62         decayChain%decayConstant(i) = lnTwo/decayChain%halfLife(i)
63     End If
64     If(decayChain%halfLife(i).le.0.0D0.and.decayChain%decayConstant(i).gt.0.0D0)Then
65 ! complete decay constant from half life
66         decayChain%halfLife(i) = lnTwo/decayChain%decayConstant(i)
67     End If
68 ! Adjust for stable isotope
69     If(decayChain%decayConstant(i).lt.0.0D0)Then
70         decayChain%halfLife(i) = -1.0D0
71         decayChain%decayConstant(i) = 0.0D0
72     End If
73     If(decayChain%halfLife(i).lt.0.0D0)Then
74         decayChain%halfLife(i) = -1.0D0
75         decayChain%decayConstant(i) = 0.0D0
76     End If
77 ! Break out if stable
78     If(decayChain%decayConstant(i).eq.0.0D0)Then
79         Exit
80     End If
81 End Do
82 decayChain%isotopes = n
83 End Subroutine decayChainComplete
84
85 Subroutine DecayBreakInfinities(L,n)
86 !
87 Implicit None ! Force declaration of all variables

```

```

88 ! Vars In/Out
89  Real(kind=DoubleReal), Dimension(:) :: L ! Lambda
90  Integer(kind=StandardInteger) :: n
91 ! Vars Private
92  Integer(kind=StandardInteger) :: i,j
93  Logical :: breaking
94 ! Loop and alter matching decay constants slightly
95  breaking = .true.
96  Do While(breaking)
97    breaking = .false.
98    Do i=1,n-1
99      Do j=i+1,n
100        If(L(i).eq.L(j))Then
101          breaking = .true.
102          L(i) = L(i)*1.0000001D0 ! Vary by 0.00001%
103        End If
104      End Do
105    End Do
106  End Do
107 End Subroutine DecayBreakInfinities
108
109 Function CalcIsotopeChainCalc(t,m,W,L,N,B) Result (nEnd)
110  Implicit None ! Force declaration of all variables
111 ! Vars In
112  Real(kind=DoubleReal) :: t
113  Integer(kind=StandardInteger) :: m
114  Real(kind=DoubleReal), Dimension(1:100) :: W ! Production Rate
115  Real(kind=DoubleReal), Dimension(1:100) :: L ! Lambda
116  Real(kind=DoubleReal), Dimension(1:100) :: N ! Starting number of atoms
117  Real(kind=DoubleReal), Dimension(1:100) :: B ! Branch factor
118 ! Vars Out
119  Real(kind=DoubleReal) :: nEnd
120 ! Vars Private
121  Integer(kind=StandardInteger) :: i, j, k, z
122  Real(kind=DoubleReal) :: mult, multR
123  Real(kind=DoubleReal) :: nChange
124 ! Init
125  nEnd = 0.0D0
126 ! -----
127 ! UNSTABLE Isotopes
128 ! -----
129  If(L(m).gt.0.0D0)Then
130 ! Loop through terms
131  Do k=1,m
132    multR = CalcIsotopeChainMultR(k,m,L,B)
133    mult = multR * N(k)
134 ! decay of starting matter
135    nChange = CalcIsotopeChainF_Unstable(k,m,t,mult,L)
136    nEnd = nEnd + nChange
137 ! production term
138    mult = multR * W(k)
139    nChange = CalcIsotopeChainG_Unstable(k,m,t,mult,L)
140    nEnd = nEnd + nChange

```

```

141      print *,k,nEnd
142      End Do
143      Else
144 ! -----
145 ! STABLE Isotopes
146 ! -----
147 ! Loop through terms
148      nEnd = N(m)+t*W(m)
149      Do k=1,m-1
150          multR = CalcIsotopeChainMultR(k,m,L,B)
151          mult = multR * N(k)
152 ! decay of starting matter
153          nChange = CalcIsotopeChainF_Stable(k,m,t,mult,L)
154          nEnd = nEnd + nChange
155 ! production term
156          mult = multR * W(k)
157          nChange = CalcIsotopeChainG_Stable(k,m,t,mult,L)
158          nEnd = nEnd + nChange
159          print *,k,nEnd
160      End Do
161      End If
162 End Function CalcIsotopeChainCalc
163
164 Function CalcIsotopeChainMultR(k,m,L,B) Result (multR)
165 ! Vars In
166     Integer(kind=StandardInteger) :: k, m
167     Real(kind=DoubleReal), Dimension(1:100) :: L ! Lambda
168     Real(kind=DoubleReal), Dimension(1:100) :: B ! Branching Factor
169 ! Vars Out
170     Real(kind=DoubleReal) :: multR
171 ! Private
172     Integer(kind=StandardInteger) :: i
173 ! Result
174     multR = 1.0D0
175     If(k.lt.m)Then
176         Do i=k,m-1
177             multR = multR * B(i+1) * L(i)
178         End Do
179     End If
180 End Function CalcIsotopeChainMultR
181
182 ! -----
183 ! Unstable Isotope Functions
184 ! -----
185
186 Function CalcIsotopeChainF_Unstable(k,m,t,mult,L) Result (nChange)
187 ! Vars In
188     Integer(kind=StandardInteger) :: k, m
189     Real(kind=DoubleReal) :: t, mult
190     Real(kind=DoubleReal), Dimension(1:100) :: L ! Lambda
191 ! Vars Out
192     Real(kind=DoubleReal) :: nChange
193 ! Private

```

```

194  Real(kind=DoubleReal) :: multP
195  Real(kind=DoubleReal) :: p, q, r, s
196  Integer(kind=StandardInteger) :: i, j
197 ! Calculate isotope amount change
198  nChange = 0.0D0
199  multP = (-1.0D0)**(m-k)
200 ! Loop through pfrac
201  Do i=k,m
202    r = 1.0D0
203    Do j=k,m
204      If(j.ne.i)Then
205        r = r * (L(i)-L(j))
206      End If
207    End Do
208    nChange = nChange + (1.0D0/r)*exp(-1.0D0*L(i)*t)*multP*mult
209  End Do
210 End Function CalcIsotopeChainF_Unstable
211
212 Function CalcIsotopeChainG_Unstable(k,m,t,mult,L) Result (nChange)
213 ! Vars In
214  Integer(kind=StandardInteger) :: k, m
215  Real(kind=DoubleReal) :: t, mult
216  Real(kind=DoubleReal), Dimension(1:100) :: L ! Lambda
217 ! Vars Out
218  Real(kind=DoubleReal) :: nChange
219 ! Private
220  Real(kind=DoubleReal) :: multP
221  Real(kind=DoubleReal) :: p, q, r, s
222  Integer(kind=StandardInteger) :: i, j
223 ! Calculate isotope amount change
224  nChange = 0.0D0
225  multP = (-1.0D0)**(m-k+1)
226 ! term A
227  r = 1.0D0
228  Do i=k,m
229    r = r * L(i)
230  End Do
231  nChange = nChange + (1.0D0/r)*mult
232 ! Loop through pfrac
233  Do i=k,m
234    r = 1.0D0*L(i)
235    Do j=k,m
236      If(j.ne.i)Then
237        r = r * (L(i)-L(j))
238      End If
239    End Do
240    nChange = nChange + (1.0D0/r)*exp(-1.0D0*L(i)*t)*multP*mult
241  End Do
242 End Function CalcIsotopeChainG_Unstable
243
244 ! -----
245 ! Stable Isotope Functions
246 ! -----

```

```

247
248     Function CalcIsotopeChainF_Stable(k,mIn,t,mult,L) Result (nChange)
249     ! Vars In
250         Integer(kind=StandardInteger) :: k, mIn
251         Real(kind=DoubleReal) :: t, mult
252         Real(kind=DoubleReal), Dimension(1:100) :: L ! Lambda
253     ! Vars Out
254         Real(kind=DoubleReal) :: nChange
255     ! Private
256         Integer(kind=StandardInteger) :: m
257         Real(kind=DoubleReal) :: multP
258         Real(kind=DoubleReal) :: p, q, r, s
259         Integer(kind=StandardInteger) :: i, j
260     ! In
261         m = mIn-1
262     ! Calculate isotope amount change
263         nChange = 0.0D0
264         multP = (-1.0D0)**(m-k+1)
265     ! term A
266         r = 1.0D0
267         Do i=k,m
268             r = r * L(i)
269         End Do
270         nChange = nChange + (1.0D0/r)*mult
271     ! Loop through pfrac
272         Do i=k,m
273             r = 1.0D0*L(i)
274             Do j=k,m
275                 If(j.ne.i)Then
276                     r = r * (L(i)-L(j))
277                 End If
278             End Do
279             nChange = nChange + (1.0D0/r)*exp(-1.0D0*L(i)*t)*multP*mult
280         End Do
281     End Function CalcIsotopeChainF_Stable
282
283
284     Function CalcIsotopeChainG_Stable(k,mIn,t,mult,L) Result (nChange)
285     ! Vars In
286         Integer(kind=StandardInteger) :: k, mIn
287         Real(kind=DoubleReal) :: t, mult
288         Real(kind=DoubleReal), Dimension(1:100) :: L ! Lambda
289     ! Vars Out
290         Real(kind=DoubleReal) :: nChange
291     ! Private
292         Integer(kind=StandardInteger) :: m
293         Real(kind=DoubleReal) :: multP
294         Real(kind=DoubleReal) :: p, q, r, s
295         Integer(kind=StandardInteger) :: i, j
296     ! In
297         m = mIn-1
298     ! Calculate isotope amount change
299         nChange = 0.0D0

```

```

300      multP = (-1.0D0)**(m-k)
301 ! term A
302      r = 1.0D0
303      Do i=k,m
304          r = r * L(i)
305      End Do
306      nChange = nChange + (1.0D0/r)*t*mult
307 ! term B
308      p = CalcIsotopeChainC(L,k,m)
309      q = 1.0D0
310      Do i=k,m
311          q = q*L(i)*L(i)
312      End Do
313      r = (-1.0D0)*(p/q)
314      nChange = nChange + r*mult
315 ! Loop through pfrac
316      Do i=k,m
317          r = 1.0D0*L(i)*L(i)
318          Do j=k,m
319              If(j.ne.i)Then
320                  r = r * (L(i)-L(j))
321              End If
322          End Do
323          nChange = nChange + (1.0D0/r)*exp(-1.0D0*L(i)*t)*multP*mult
324      End Do
325 End Function CalcIsotopeChainG_Stable
326
327 Function CalcIsotopeChainC(L,k,m) Result (numerator)
328 ! Calculates numerator in isotope activity function
329 Implicit None ! Force declaration of all variables
330 ! Vars In
331     Real(kind=DoubleReal), Dimension(:) :: L
332     Integer(kind=StandardInteger) :: k, m
333 ! Vars Out
334     Real(kind=DoubleReal) :: numerator
335 ! Vars Private
336     Integer(kind=StandardInteger) :: i, j
337     Real(kind=DoubleReal) :: tempMult
338     numerator = 0.0D0
339     Do i=k,m
340         tempMult = 1.0D0
341         Do j=k,m
342             If(j.ne.i)Then
343                 tempMult = tempMult * L(j)
344             End If
345         End Do
346         numerator = numerator + tempMult
347     End Do
348 End Function CalcIsotopeChainC

```

Bibliography

- [1] A. J. Koning D. Rochman. “Modern Nuclear Data Evaluation With The TALYS Code System”. In: *Nuclear Data Sheets* 113 (2012).
- [2] J. P. Biersack James F. Ziegler M. D. Ziegler. “SRIM - The stopping and range of ions in matter”. In: *Nuclear Instruments and Methods in Physics Research Section B* 268 (2010), pp. 1818–1823.
- [3] A. A. Sonzogni. *Nuclear Data Services*. www-nds.iaea.org/ndspub/download-endf/ENDF-B-VII.0/decay-index.htm (visited on 08/14/2015).
- [4] M. Herman M. B. Chadwick. “ENDF/B-VII.0: Next generation evaluated nuclear data library for nuclear science and technology.” In: *Nucl. Data Sheets* 107 (2006), p. 2931.
- [5] P. Blaise M. Coste A. Courcelle T.D. Huynh C. Jouanne P. Leconte O. Litaize S. Mengelle G. Nogure J-M. Ruggiri O. Srot J. Tommasi C. Vaglio J-F. Vidal A. Santamarina D. Bernard. *The JEFF-3.1.1 Nuclear Data Library*. 2009. ISBN: 978-92-64-99074-6.
- [6] H. Stehfest. “Algorithm 368: Numerical Inversion of Laplace Transform”. In: *Communications of the ACM* 13 (1970), pp. 47–49.
- [7] J. D. Hunter. “Matplotlib: A 2D graphics environment”. In: *Computing In Science & Engineering* 9 (2007), pp. 90–95.

C.2 Activity V2

C.2.1 Accompanying Manual

UNIVERSITY OF BIRMINGHAM



Department of Metallurgy & Materials

Activity V2 Manual

Contents

1 Overview	2
1.1 Reason for the Program	2
1.2 Program Requirements	2
2 Installation	3
3 How To Use	4
3.1 SRIM	4
3.2 Activity	5
3.3 Input Keywords	11
4 Examples	12
4.1 Iron - 28MeV Protons	12
4.2 Iron Fe56 (only) - 28MeV Protons	12
4.3 Steel - 5MeV Protons	12
5 Decay Equation	14
5.1 Bateman Equation	14
5.1.1 Laplace Transform	14
5.1.2 Constructing the Differential Equations	15
5.1.3 Numerical Inversion of the Laplace Transform	15
5.1.4 Analytic Solution by Partial Fraction Expansion	16
5.1.5 Preference: Analytic over Numeric	18
5.2 Python Isotopes Class	19
6 Future Plans	22
6.1 Selection of Cross Section Databases	22

Chapter 1

Overview

1.1 Reason for the Program

The Activity program calculates how radioactive a target becomes after being irradiated by high energy ions. It uses the TENDL-2019[1] database that contains cross-section data for protons and deuterons, and the JEFF-3.3 Radioactive Decay Data File[2].

1.2 Program Requirements

The user must provide a exyz file from the SRIM[3] ion transport program and a breakdown of the composition of the target, as well as other irradiation parameters (beam projectile, beam duration, beam area, target thickness, target composition, simulation end time etc).

Chapter 2

Installation

The program needs Python 3 installed in order to run. At the time of writing, it has only been developed to run on a Linux operating system, but it shouldn't require much adjusting to run on a Windows computer too.

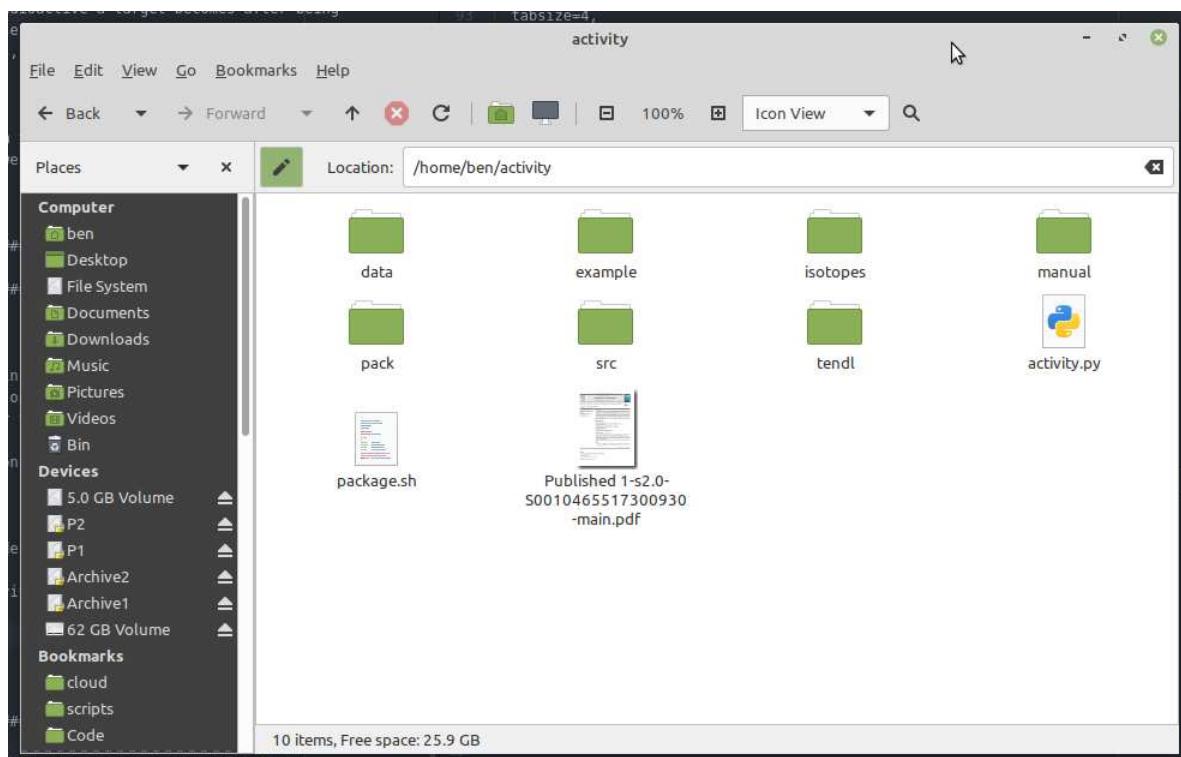
Download and install the latest version of Python 3:

<https://www.python.org/downloads>

The latest version of the activity code with data files must also be downloaded:

https://github.com/BenPalmer1983/activity_v2

For example, I have installed to /home/ben/activity

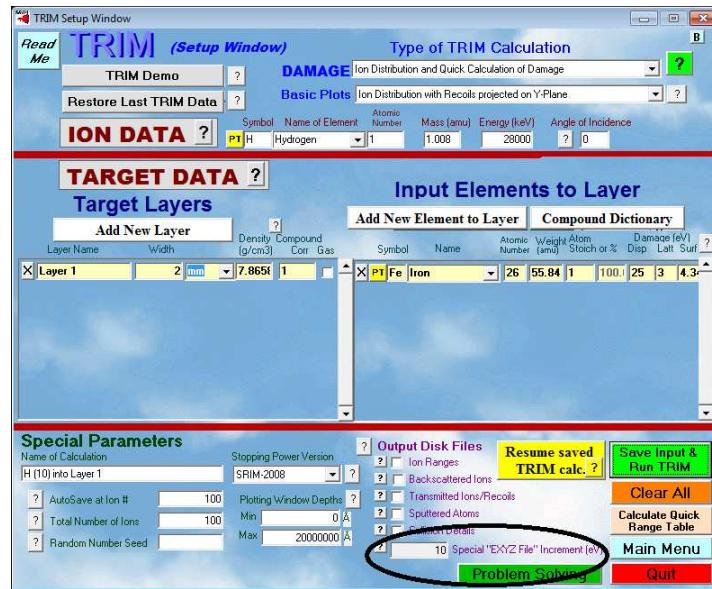


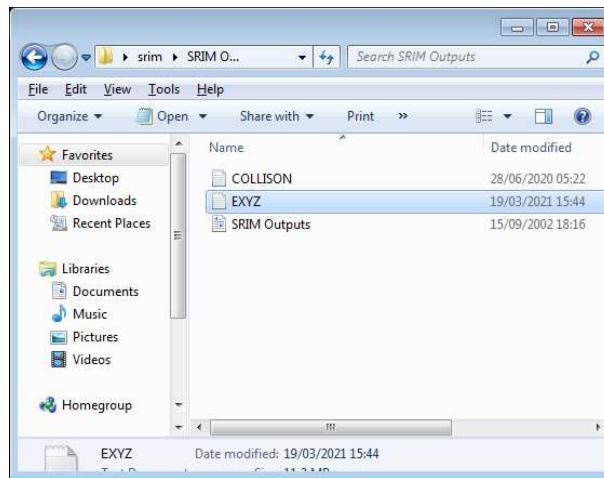
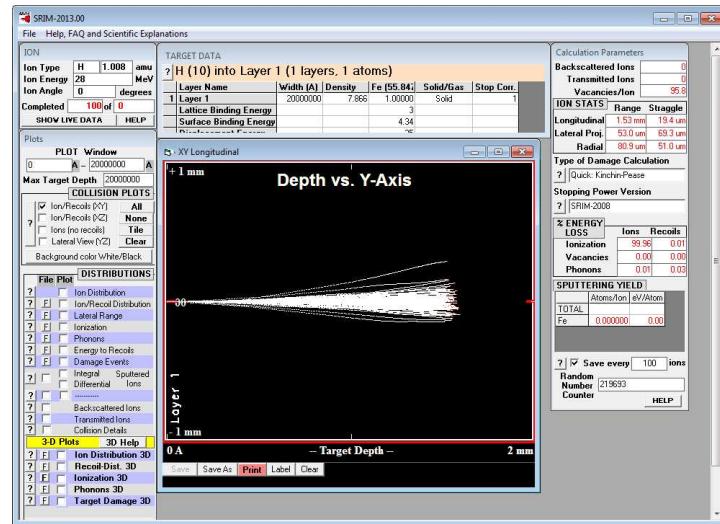
Chapter 3

How To Use

3.1 SRIM

If you haven't done so already, run the required simulation in SRIM. When setting up the calculation, be sure to set an increment for the EXYZ file. Sane values that have been used in examples range from 10eV to 10,000KeV, but this will depend on the resolution of the data in the TENDL database, the thickness of your target and the energy of the projectiles.

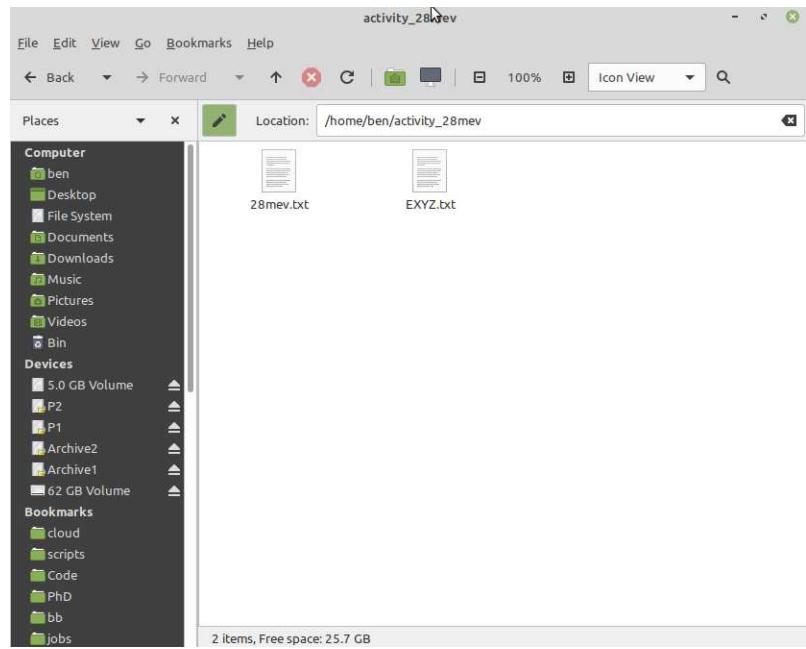




Take a copy of the EXYZ.txt file as it will be required by the activity program.

3.2 Activity

Create a directory to run the calculation. Copy in the EXYZ.txt file and create an input file.



The input file is just a text file, and will specify the calculation details. The data file paths must be specified, and then the simulations can be detailed, each with it's own unique name. The example below is for two simulations - the first with a 300s beam, and the second with a 3000s beam.

```

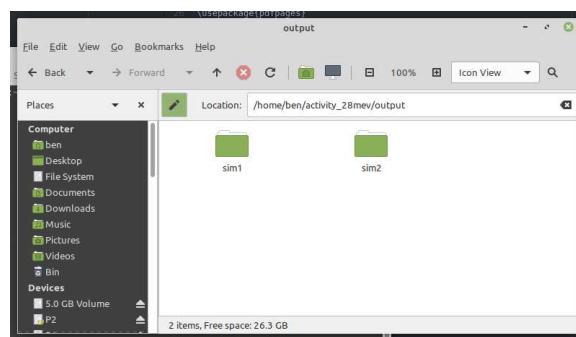
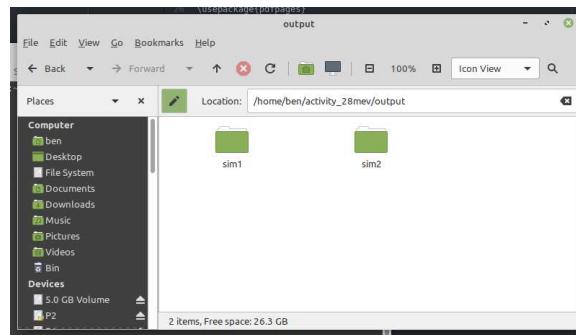
1 # Data files
2 data_isotopes="/home/ben/activity/data/isotopes" xs="/home/ben/activity/data/xs"
3
4 # Sim 1
5 sim1 exyz="EXYZ.txt" target_composition=Fe,100.0 target_depth=0.5,mm target_density=7808,kgm3
   beam_projectile='proton' beam_energy=28,MeV beam_area=64,mm2 beam_duration=300,s beam_current=0.5,
   uA end_time=260000,s
6
7 # Sim 2
8 sim2 exyz="EXYZ.txt" target_composition=Fe,100.0 target_depth=0.5,mm target_density=7808,kgm3
   beam_projectile='proton' beam_energy=28,MeV beam_area=64,mm2 beam_duration=3000,s beam_current
   =5.0,uA end_time=260000,s

```

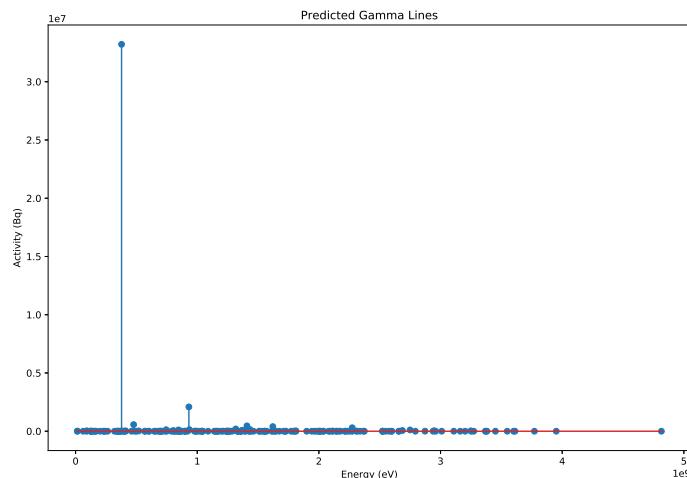
The program is run from the terminal/command line. Depending on your computer system, python3 might run through the command python3 or python, and on my computer the command is python3.

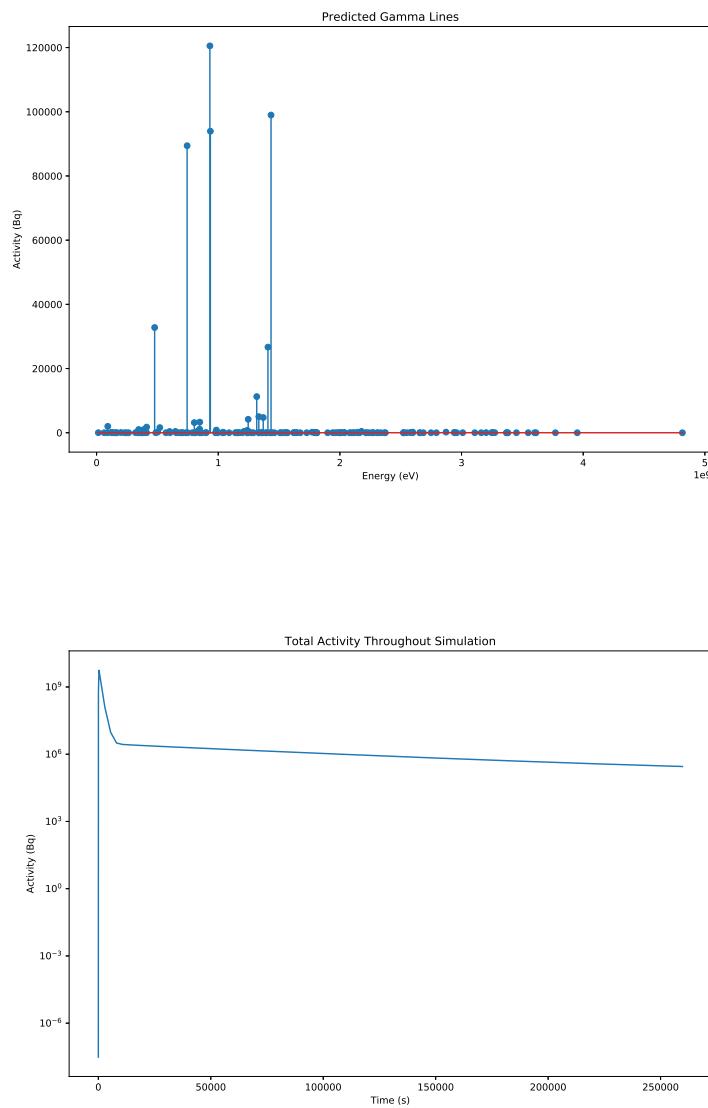
```
1 python3 /home/ben/activity/activity.py /home/ben/activity_28mev/28mev.txt
```

The program outputs into a directory for each simulation.



Within each directory are various plots and data files, including the activity over time and predicted gamma lines.





The program analyses the exyz file and output a chart showing the distribution of energy lost by projectiles as they pass through the target.

As a target is irradiated, the residual radioactive isotopes begin to decay. To begin with, the production outweighs the decay, but at a certain point there's enough of the radioactive isotope within the target that the decay balances with the production. At this point the radioactive isotope in the target is saturated, and will not increase above this amount unless the beam parameters are adjusted.

The saturation times depend on the isotope and the various reaction cross sections with the projectile. The saturation plots for each radioactive isotope are created and saved by the program.

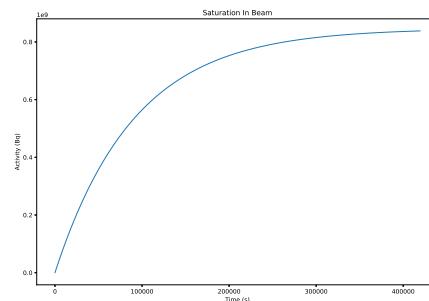
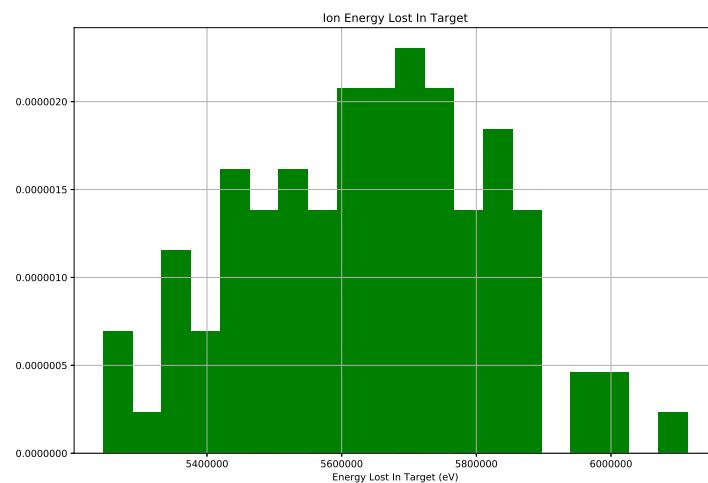


Figure 3.1: Saturation of Cobalt-55

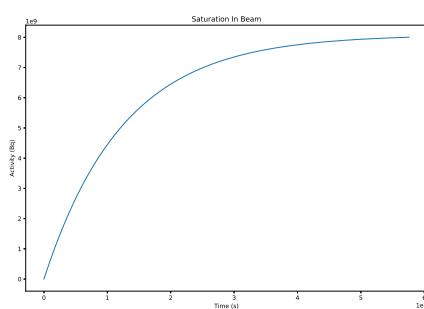


Figure 3.2: Saturation of Iron-55

The program calculates the dose a human may receive from the target at the time it is removed from the beam, and at the time the simulation stops.

Listing 3.1: Predicted Dose

```

1 Gamma Dose - Beam End
2 =====
3
4 Activity/Bq           54.4306601689
5 Power eV/s            28342871818.2
6 Power J/s             4.54103823697e-09
7 Dose Gy/s             4.51575003676e-12
8 Dose Gy/hr             1.62567001323e-08
9 Percentage of annual dose/hr 0.0142506237521

10
11 Gamma Dose - Sim End
12 =====
13
14 Activity/Bq           5.80221584491
15 Energy eV/s            2227911995.02
16 Energy J/s             3.56951604018e-10
17 Dose Gy                8.52319971389e-13
18 Dose Gy/hr              1.27786970412e-09
19 Percentage of annual dose/hr 0.00112018061534

20
21
22 Absorbed Dose Calculations
23 =====
24 Absorbed dose assumptions:
25 1. radiation from point, emitted isotropically
26 2. 80Kg human
27 3. 1m from point source
28 4. 1m squared surface area
29 5. all energy absorbed

30
31
32 Dose Limits
33 =====
34 employees 18+          20 millisieverts/year
35 trainees 18+           6 millisieverts/year
36 public and under 18s    1 millisievert/year
37 public and under 18s    1.140771128E-04 millisieverts/hour

38
39 Dose averaged over area of skin not exceeding 1cm2
40 Source: http://www.hse.gov.uk/radiation/ionising/doses/

```

3.3 Input Keywords

Command	<i>Sub Command</i>	Description
data		data fields are input here
data	isotopes	path to isotopes directory
data	xs	path to cross section directory
sim1		each simulation is numbered sequentially from 1 i.e. sim1 sim2 sim3...
sim1	exyz	path to exyz SRIM file
sim1	target_composition	isotopes that make up the beam target
sim1	target_depth	the depth of material through which the beam passes
sim1	target_density	how dense (specify kgm3 or GCM3)
sim1	beam_projectile	currently, only "proton", but this could be extended in the future
sim1	beam_energy	energy of the protons/projectiles
sim1	beam_area	how much area the beam covers
sim1	beam_duration	how long the beam will be on for (specify s,minute,hour) t=0 to beam_duration is the first part of the calculation
sim1	end_time	the time the calculation and final activity is measured after t=0 (specify s,minute,hour) end_time to beam_duration is the second part of the calculation

Chapter 4

Examples

4.1 Iron - 28MeV Protons

In this example, natural iron makes up the target .

```

1 # Data files
2 data isotopes="/home/ben/activity/data/isotopes" xs="/home/ben/activity/data/xs"
3
4 # Sim
5 sim1 exyz="XYZ.txt" target_composition=Fe,100.0 target_depth=0.5,mm target_density=7808,kgm3
   beam_projectile='proton' beam_energy=28,MeV beam_area=64,mm2 beam_duration=300,s beam_current=0.5,
   uA end_time=260000,s

```

The simulation section defines what the target material is made of as well as the beam parameters.

4.2 Iron Fe56 (only) - 28MeV Protons

In this example, the target is made of Fe56 only, and contains none of the other naturally occurring stable isotopes.

```

1 # Data files
2 data isotopes="/home/ben/activity/data/isotopes" xs="/home/ben/activity/data/xs"
3
4 # Sim
5 sim1 exyz="XYZ.txt" target_composition=Fe56,100.0 target_depth=0.5,mm target_density=7808,kgm3
   beam_projectile='proton' beam_energy=28,MeV beam_area=64,mm2 beam_duration=300,s beam_current=0.5,
   uA end_time=260000,s

```

4.3 Steel - 5MeV Protons

This example was provided by Alex Dickinson-Lomas, with a steel containing a wider range of elements.

```

1 # Data files
2 data isotopes="/home/ben/activity/data/isotopes" xs="/home/ben/activity/data/xs"
3

```

```
4 # Sim
5 sim1 exyz="EXYZ.txt" target_composition=Fe,96.375,C,0.772,Cu,0.024,Mn,1.36,Ni,0.698,Si,0.381,Cr,0.092,
V,0.008,P,0.009,Si,0.003,Mo,0.278 target_depth=0.1,mm target_density=7808,kgm3 beam_projectile='
proton' beam_energy=5,MeV beam_area=64,mm2 beam_duration=300,s beam_current=0.5,uA end_time
=260000,s
```

Chapter 5

Decay Equation

5.1 Bateman Equation

The Bateman equation was derived using Laplace transforms, and this same method has been used to develop a modified equation that incorporates branching factors and production rates for each isotope in the decay chain, as illustrated by Figure 5.1.

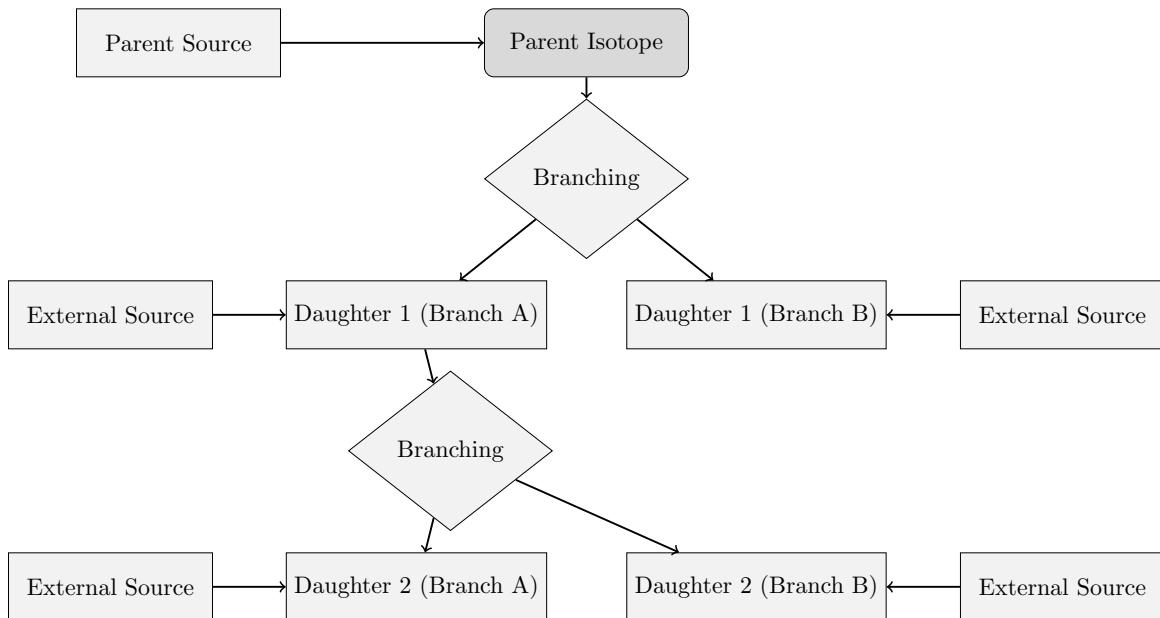


Figure 5.1: An example of several decay chains including branching factors and possible external source terms for each isotope on each chain.

5.1.1 Laplace Transform

Laplace Transforms (5.1) are a useful mathematical tool, and allow ordinary differential equations to be solved by simple algebraic manipulation in the s domain. Bateman took advantage of Laplace Transforms in deriving his equation, and this is the method that has been taken here as well.

$$F(s) = \int_0^{\infty} f(t) \exp(-st) dt \quad (5.1)$$

5.1.2 Constructing the Differential Equations

The first step is to set up differential equations for the parent isotope, unstable daughter isotopes and stable daughter isotope. The parent isotope has a source term, due to production, and a loss term, due to decay. The unstable daughter isotopes have two source terms, from the production by irradiation induced transmutation and the decay of preceding isotopes in the decay chain, and a loss term, due to decay. Finally, the stable daughter that finalizes the decay chain has two source terms (the same as the unstable daughters) but no loss term.

The variables (and vectors) used in these equations are defined as follows:

- $\vec{\lambda}$ vector containing isotope decay constants λ_i
- \vec{b} vector containing isotope to isotope branching factors b_i
- \vec{w} vector containing isotope production rates w_i
- t time at which activity/amount of isotope is measured
- $N_i(0)$ starting amount of the i^{th} isotope
- $N_i(t)$ amount of the i^{th} isotope at time t
- $N'_i(t)$ change in amount of the i^{th} isotope, with respect to time, at time t

The differential equations for the parent isotope (first isotope), unstable daughter isotopes (i^{th} isotopes) and stable, final, daughter isotope (z^{th} isotope) in the time domain are as follows:

$$N'_1(t) = \omega_1 - \lambda_1 N_1(t) \quad (5.2)$$

$$N'_i(t) = \omega_i + b_{i-1} \lambda_{i-1} N_{i-1}(t) - \lambda_i N_i(t) \quad (5.3)$$

$$N'_z(t) = \omega_z + b_{z-1} \lambda_{z-1} N_{z-1}(t) \quad (5.4)$$

Applying the Laplace Transform to these three differential equations allows them to be manipulated and solved algebraically in the s-domain.

$$N_1(s) = \frac{1}{s + \lambda_1} N_1(0) + \frac{1}{s(s + \lambda_1)} \omega_1 \quad (5.5)$$

$$N_i(s) = \frac{1}{s(s + \lambda_i)} (\omega_i) + \frac{1}{s + \lambda_i} (b_{i-1} \lambda_{i-1} N_{i-1}(s)) + \frac{1}{s + \lambda_i} N_i(0) \quad (5.6)$$

$$N_z(s) = \frac{1}{s^2} \omega_z + \frac{1}{s} b_{z-1} \lambda_{z-1} N_{z-1}(s) + \frac{1}{s} N_z(0) \quad (5.7)$$

5.1.3 Numerical Inversion of the Laplace Transform

The Gaver-Stehfest[4] algorithm was developed in the 1960s and 1970s and is a method of calculating the inverse of a Laplace Transform in the real number domain. It is an easy to implement and reasonably accurate method,

although it is an approximation to the real value. A comparison between an analytic and numeric inversion for the unstable isotope Po-218 is discussed at the end of this section (figure 5.2).

$$f(t) \approx f_n(t) = \frac{\ln(2)}{t} \sum_{k=1}^{2n} a_k(n) F(s) \text{ where } n \geq 1, t > 0 \quad (5.8)$$

$$s = \frac{k \ln(2)}{t} \quad (5.9)$$

$$a_k(n) = \frac{(-1)^{(n+k)}}{n!} \sum_{j=\text{Floor}(\frac{k+1}{2})} j^{n+1} \binom{n}{j} \binom{2j}{j} \binom{j}{k-j} \quad (5.10)$$

The equation for the i^{th} isotope may be calculated by recursively calculating the equations by numeric inversion, starting from the first (parent isotope) and inserting the result into each subsequent recursion until the i^{th} isotope is reached (changing the equations appropriately for the parent, unstable daughter and stable daughter isotopes).

5.1.4 Analytic Solution by Partial Fraction Expansion

The equation for the i^{th} isotope in the s domain can be written in full by substituting the preceding equation until the parent isotope is reached, and this full equation may be rearranged with the production amount of each isotope and starting amount of each isotope in individual terms. Each of these terms is multiplied by a fraction that can be expanded, using partial fractions, and inverted analytically.

This is illustrated with an example unstable isotope, fourth in the decay chain (including the parent isotope):

$$\begin{aligned} N_4(s) = & \frac{1}{(s + \lambda_1)(s + \lambda_2)(s + \lambda_3)(s + \lambda_4)} b_2 b_3 b_4 \lambda_1 \lambda_2 \lambda_3 N_1(0) \\ & + \frac{1}{(s + \lambda_2)(s + \lambda_3)(s + \lambda_4)} b_3 b_4 \lambda_2 \lambda_3 N_2(0) \\ & + \frac{1}{(s + \lambda_3)(s + \lambda_4)} b_4 \lambda_3 N_3(0) \\ & + \frac{1}{(s + \lambda_4)} N_4(0) \quad (5.11) \\ & + \frac{1}{s(s + \lambda_1)(s + \lambda_2)(s + \lambda_3)(s + \lambda_4)} b_2 b_3 b_4 \lambda_1 \lambda_2 \lambda_3 \omega_1 \\ & + \frac{1}{s(s + \lambda_2)(s + \lambda_3)(s + \lambda_4)} b_3 b_4 \lambda_2 \lambda_3 \omega_2 \\ & + \frac{1}{s(s + \lambda_3)(s + \lambda_4)} b_4 \lambda_3 \omega_3 \\ & + \frac{1}{s(s + \lambda_4)} \omega_4 \end{aligned}$$

An example stable isotope, fourth (last) in the decay chain (including the parent isotope):

$$\begin{aligned}
N_4(s) = & \frac{1}{s(s + \lambda_1)(s + \lambda_2)(s + \lambda_3)} b_2 b_3 b_4 \lambda_1 \lambda_2 \lambda_3 N_1(0) \\
& + \frac{1}{s(s + \lambda_2)(s + \lambda_3)} b_3 b_4 \lambda_2 \lambda_3 N_2(0) \\
& + \frac{1}{s(s + \lambda_3)} b_4 \lambda_3 N_3(0) \\
& + N_4(0) \\
& + \frac{1}{s^2(s + \lambda_1)(s + \lambda_2)(s + \lambda_3)} b_2 b_3 b_4 \lambda_1 \lambda_2 \lambda_3 \omega_1 \\
& + \frac{1}{s^2(s + \lambda_2)(s + \lambda_3)} b_3 b_4 \lambda_2 \lambda_3 \omega_2 \\
& + \frac{1}{s^2(s + \lambda_3)} b_4 \lambda_3 \omega_3 \\
& + \frac{1}{s^2} \omega_4
\end{aligned} \tag{5.12}$$

By using partial fraction expansion and standard Laplace Transforms, the set of equations below is used to calculate the amount of the m^{th} isotope in the decay chain, providing the m^{th} isotope is unstable.

$$N_m(t; \vec{\lambda}, \vec{b}, \vec{w}) = \sum_{k=1, m} r(k; \vec{\lambda}, \vec{b}) [f(t; k, m, \vec{\lambda}) N_k(0) + g(t; k, m, \vec{\lambda}) w_k] \tag{5.13}$$

$$r(k, m, \vec{\lambda}) = \begin{cases} \prod_{i=k, m-1} (b_{i+1} \lambda_i), & \text{if } k < m \\ 1, & \text{if } k = m \end{cases} \tag{5.14}$$

$$f(t; k, m, \vec{\lambda}) = (-1)^{m-k} \sum_{i=k, m} \left[\exp(-\lambda_i t) \prod_{j=k, m; j \neq i} \left(\frac{1}{\lambda_i - \lambda_j} \right) \right] \tag{5.15}$$

$$g(t; k, m, \vec{\lambda}) = \frac{1}{\prod_{i=k, m} \lambda_i} + (-1)^{m-k+1} \sum_{i=k, m} \left[\frac{1}{\lambda_i} \exp(-\lambda_i t) \prod_{j=k, m; j \neq i} \left(\frac{1}{\lambda_i - \lambda_j} \right) \right] \tag{5.16}$$

The set of equations below is used to calculate the amount of the m^{th} isotope in the decay chain, where the m^{th} isotope is stable.

$$N_m(t; \vec{\lambda}, \vec{b}, \vec{w}) = N_m + w_m t + \sum_{k=1, m-1} r(k; \vec{\lambda}, \vec{b}) [f(t; k, m-1, \vec{\lambda}) N_k(0) + g(t; k, m, \vec{\lambda}) w_k] \tag{5.17}$$

$$r(k, m, \vec{\lambda}) = \begin{cases} \prod_{i=k, m-1} (b_{i+1} \lambda_i), & \text{if } k < m \\ 1, & \text{if } k = m \end{cases} \tag{5.18}$$

$$f(t; k, m, \vec{\lambda}) = \frac{1}{\prod_{i=k,m} \lambda_i} + (-1)^{m-k+1} \sum_{i=k,m} \left[\frac{1}{\lambda_i} \exp(-\lambda_i t) \prod_{j=k,m; j \neq i} \left(\frac{1}{\lambda_i - \lambda_j} \right) \right] \quad (5.19)$$

$$g(t; k, m, \vec{\lambda}) = \frac{1}{\prod_{i=k,m} \lambda_i} t + \frac{\sum_{i=k,m} \left[\prod_{j=k,m; j \neq i} \lambda_j \right]}{\prod_{i=k,m} \lambda_i^2} + (-1)^{m-k+1} \sum_{i=k,m} \left[\frac{1}{\lambda_i^2} \exp(-\lambda_i t) \prod_{j=k,m; j \neq i} \left(\frac{1}{\lambda_i - \lambda_j} \right) \right] \quad (5.20)$$

5.1.5 Preference: Analytic over Numeric

The numeric solution only requires the equation to be solved in the s-domain; the Gaver-Stehfest algorithm performs the inversion. It is worth the extra effort to derive and implement an analytic solution, as the numeric is only an approximation. Examples of the pitfalls of the numeric solution are that it can give negative amounts of an isotope and the difference between the numeric and analytic calculated amounts can become quite large when the isotope decays away to a very small value. Figure 5.2 shows the predicted decay of a sample of Po-218 irradiated for 1,000s, and sampled until 10,000s. In the region between 4,000s and 9,000s the amount from the numeric calculation drops below zero, whereas the analytic calculation remains above zero, as would be expected.

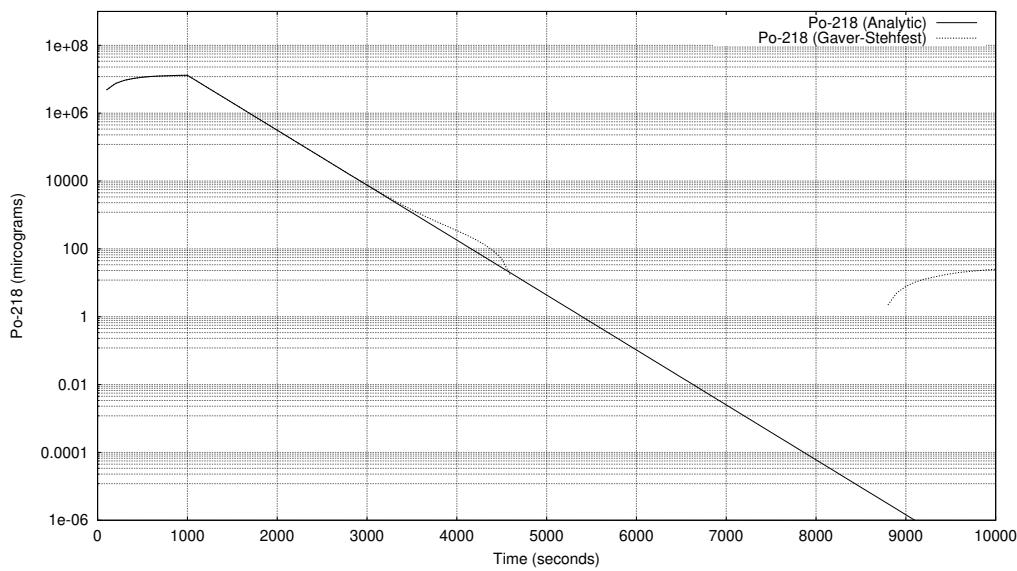


Figure 5.2: Decay of Po-218: Analytic and Gaver-Stehfest Calculations [jeff311]

5.2 Python Isotopes Class

The decay equations are computed within the isotopes class in the isotopes.py file.

```

1   class isotopes:
2
3   #####
4   # DECAY EQUATIONS
5   #####
6
7
8   @staticmethod
9   def calculate_activity(t, l, b, w, n0):
10  nt = numpy.zeros((len(n0),))
11  for m in range(0,len(n0)):
12      if(l[m] > 0.0):
13          nt[m] = isotopes.activity_unstable(t, l, b, w, n0, m)
14      elif(l[m] == 0.0):
15          nt[m] = isotopes.activity_stable(t, l, b, w, n0, m)
16  return nt
17
18 @staticmethod
19 def activity_unstable(t, l, b, w, n0, m):
20     s = 0.0
21     for k in range(0, m+1):
22         s = s + isotopes.r(k, m, b, l) * ( isotopes.f_unstable(t,k,m,l) * n0[k] + isotopes.g_unstable(t,
23                                         k,m,l) * w[k])
24
25 @staticmethod
26 def activity_stable(t, l, b, w, n0, m):
27     s = n0[m] + w[m] * t

```

```

28     for k in range(0, m):
29         s = s + isotopes.r(k, m, b, l) * (isotopes.f_stable(t,k,m,l) * n0[k] + isotopes.g_stable(t,k,m,l)
30                                     ) * w[k])
31     return s
32
33 @staticmethod
34 def r(k, m, b, l):
35     if(k == m):
36         return 1.0
37     else:
38         p = 1.0
39         for i in range(k, m):
40             p = p * (b[i] * l[i])
41         return p
42
43 @staticmethod
44 def f_unstable(t,k,m,l):
45     s = 0.0
46     for i in range(k, m+1):
47         p = 1.0
48         for j in range(k, m+1):
49             if(i != j):
50                 p = p * (1 / (l[i] - l[j]))
51             s = s + numpy.exp(-1 * l[i] * t) * p
52     s = (-1)**(m-k) * s
53     return s
54
55 @staticmethod
56 def g_unstable(t,k,m,l):
57     pa = 1.0
58     for i in range(k,m+1):
59         pa = pa * l[i]
60     pa = 1.0 / pa
61     s = 0.0
62     for i in range(k, m+1):
63         pb = 1.0
64         for j in range(k, m+1):
65             if(i != j):
66                 pb = pb * (1 / (l[i]-l[j]))
67             s = s + (1/l[i]) * numpy.exp(-l[i]*t) * pb
68     return pa + s * (-1)**(m-k+1)
69
70 @staticmethod
71 def f_stable(t,k,m_in,l):
72     m = m_in - 1
73
74     p = 1.0
75     for i in range(k, m+1):
76         p = p * l[i]
77
78     s = 0.0
79     for i in range(k, m+1):

```

```

80     r = l[i]
81     for j in range(k, m+1):
82         if(i != j):
83             r = r * (l[i] - l[j])
84         s = s + (1/r)*numpy.exp(-l[i]*t)
85
86     return (1.0/p) + s * (-1.0)**(m-k+1)
87
88
89 @staticmethod
90 def g_stable(t,k,m_in,l):
91     m = m_in - 1
92
93     pa = 1.0
94     for i in range(k,m+1):
95         pa = pa * l[i]
96     pa = 1.0 / pa
97
98     sa = 0.0
99     for i in range(k, m+1):
100        pb = 1.0
101        for j in range(k,m+1):
102            if(j != i):
103                pb = pb * l[j]
104            sa = sa + pb
105        pc = 1.0
106        for i in range(k, m+1):
107            pc = pc * l[i]**2
108
109        sb = 0.0
110        for i in range(k, m+1):
111            pd = 1.0
112            for j in range(k, m+1):
113                if(i != j):
114                    pd = pd * (1 / (l[i]-l[j]))
115            sb = sb + (1/(l[i]**2)) * numpy.exp(-l[i]*t) * pd
116
117    return pa * t + sa / pc + sb * (-1)**(m-k+1)

```

Chapter 6

Future Plans

6.1 Selection of Cross Section Databases

The Scanditronix MC40 Cyclotron at the University of Birmingham has several beamlines and is capable of accelerating protons, deuterons, Helium 3 and Helium 4 and fluxes and energy ranges detailed below.

Particle	Energy (MeV)	Max Current (micro A)	Flux (ions per second)
p	8-40	60	3.75×10^{14}
d	8-40	30	1.87×10^{14}
$^4He^{2+}$	8-53	30	9.36×10^{13}
$^3He^{2+}$	4-20	60	1.87×10^{14}

Table 6.1: Beam Characteristics of the Scanditronix MC-40

The current data file is for Protons, and the range of energies (for Fe-54 and Fe-56 at the very least) only range up to 30MeV. Previous versions of the TENDL data files have covered larger ranges (TENDL-2009 extended up to 200MeV).

As the energies do not cover the full range of possible energies for our own cyclotron, it would be desirable to use the TALYS program to calculate the reaction cross sections for a range up to at least 100MeV (for Protons, Deuterons, Helium 3 and Helium 4 ions) and save this cross section data into an alternate database.

Bibliography

- [1] A.J. Koning et al. “TENDL: Complete Nuclear Data Library for Innovative Nuclear Science and Technology”. In: *Nuclear Data Sheets* 155 (2019).
- [2] A.J.M. Plompen et al. “The JEFF-3.3 Nuclear Data Library”. In: *Eur. Phys. J. A* 56 (181 2020).
- [3] J. P. Biersack James F. Ziegler M. D. Ziegler. “SRIM - The stopping and range of ions in matter”. In: *Nuclear Instruments and Methods in Physics Research Section B* 268 (2010), pp. 1818–1823.
- [4] H. Stehfest. “Algorithm 368: Numerical Inversion of Laplace Transform”. In: *Communications of the ACM* 13 (1970), pp. 47–49.

Appendix D

Activity V1 Full Results

D.1 36MeV Proton Irradiated Iron

Fe irradiated by 36MeV protons: output file produced by Activity V1: <https://github.com/BenPalmer1983/activity>

Listing D.1: Isotope Activity at End of Simulation

```
1 =====
2 ACTIVITY code University of Birmingham
3 =====
4
5 Compiled: 15:08:41 11/01/2016
6 MPI Processes: 1
7
8 Date: 21:21 24/06/2016
9 Read user input          0.0000
10 Read Isotope Data       0.0000
11 Read Activity User Input File 0.0720
12 Read Trajectory File    0.0720
13 Data point count: 100887 0.9560
14
15 XS Data Files Loaded:
16 Reading xs file 1_1-26_54_0.dat
17 Reading xs file 1_1-26_56_0.dat
18 Reading xs file 1_1-26_57_0.dat
19 Reading xs file 1_1-26_58_0.dat
20
21 Calculation Settings
22
23 Beam Duration/s:      0.3000000000D+03
24 Beam Current/uA:       0.5000000000D+00
25 Beam Energy/MeV:       0.3600000000D+02
26 Beam Area/mm2:        0.1000000000D+03
27 Target Thickness/Angstrom: 0.5000000000D+07
28 Target Density/kgm-3:   0.8000000000D+04
29 Activity Measurement Time/s: 0.2600000000D+06
30 Input File:            /home/ben/activity/examples/Fe36MeV.in
31 Isotope File:          /home/ben/activity/data/isotopes.txt
32 Decay Modes File:     /home/ben/activity/data/decaymodes.txt
33 Gamma Energies File:  /home/ben/activity/data/gammaenergies.txt
34 XS File Directory:    /home/ben/activity/data/xs
35
36
37 Create material tally   1.2640
```

39 Fill tally with blank data 1.2640
 40 Fill tally with isotope data 1.2640
 41
 42 -----
 43 Starting Isotope Tally - Input Material
 44 Element Z A M mk sim Half life Decay Constant Activity Reaction Rate Atoms/mg
 45 -----
 46 FE 026 054 0 000 001 -0.1000000000D+01 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00 0.5405383878D+28
 47 FE 026 056 0 000 001 -0.1000000000D+01 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00 0.7890272999D+29
 48 FE 026 057 0 000 001 -0.1000000000D+01 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00 0.1758807969D+28
 49 FE 026 058 0 000 001 -0.1000000000D+01 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00 0.2263810953D+27
 50 Total starting isotopes: 4
 51
 52
 53 -----
 54 Pre-Simulation Summary
 55 1.2840
 56 -----
 57
 58 Beam flux ions/s: 0.3120754681E+13
 59
 60
 61 Calculate projectile-target-product reaction rates 1.2840
 62 Reaction rates to calculate: 424
 63 Trajectory fit equation:
 64 0 0.3665497627E+05
 65 1 -0.1783622595E-02
 66 2 0.2947794093E-09
 67 3 -0.3735372827E-16
 68 4 0.1919778602E-23
 69 5 -0.3566818320E-31
 70 Affected material depth: 0.5000000000E+07
 71
 72 -----
 73 Calculated variables
 74 -----
 75 Beam flux ions/s: 0.3120754681E+13
 76 Affected depth/m: 0.5000000000E-03
 77 Affected depth/ang: 0.5000000000E+07
 78 Affected volume/m³: 0.500000054E-07
 79 Number Density: 0.8626809746E+29
 80 VPI: 0.6020000000E+02
 81 DPA: 0.1306643605E-04
 82
 83 -----
 84 Reaction rates in the target material 1.2840
 85 Sym Z A M Sym Z A M Reaction Rate Average XS Number Density
 86 -----
 87 1: FE 26 54 0 NN 0 1 0 0.3508066664E+09 0.4160433001E-28 0.5403805016E+28
 88 2: FE 26 54 0 H 1 1 0 0.1351444089E+10 0.1602761044E-27 0.5403805016E+28
 89 3: FE 26 54 0 H 1 2 0 0.3205323639E+08 0.3801391343E-29 0.5403805016E+28
 90 4: FE 26 54 0 H 1 3 0 0.7165361660E+06 0.8497845101E-31 0.5403805016E+28
 91 5: FE 26 54 0 HE 2 3 0 0.8617826538E+07 0.1022041294E-29 0.5403805016E+28
 92 6: FE 26 54 0 HE 2 4 0 0.5675883691E+08 0.6731381151E-29 0.5403805016E+28
 93 67: FE 26 54 0 TI 22 46 0 0.4687831440E+01 0.5559588940E-36 0.5403805016E+28
 94 76: FE 26 54 0 V 23 47 0 0.3452942314E+06 0.4095057628E-31 0.5403805016E+28
 95 78: FE 26 54 0 V 23 49 0 0.5791127831E+07 0.6868056297E-30 0.5403805016E+28
 96 85: FE 26 54 0 CR 24 49 0 0.2416421986E+07 0.2865784131E-30 0.5403805016E+28
 97 86: FE 26 54 0 CR 24 50 0 0.4290008294E+08 0.5087785892E-29 0.5403805016E+28

98	87:	FE	26	54	0	CR	24	51	0	0.2135062196E+07	0.2532102172E-30	0.5403805016E+28
99	88:	FE	26	54	0	CR	24	52	0	0.8659953579E+08	0.1027037400E-28	0.5403805016E+28
100	92:	FE	26	54	0	MN	25	50	0	0.1542445948E+07	0.1829281950E-30	0.5403805016E+28
101	93:	FE	26	54	0	MN	25	51	0	0.1778372502E+07	0.2109081827E-30	0.5403805016E+28
102	94:	FE	26	54	0	MN	25	52	0	0.9863724488E+08	0.1169800030E-28	0.5403805016E+28
103	95:	FE	26	54	0	MN	25	53	0	0.2232277074E+09	0.2647395303E-28	0.5403805016E+28
104	99:	FE	26	54	0	FE	26	52	0	0.1143967233E+08	0.1356701421E-29	0.5403805016E+28
105	100:	FE	26	54	0	FE	26	53	0	0.1888610876E+09	0.2239820325E-28	0.5403805016E+28
106	101:	FE	26	54	0	FE	26	54	0	0.1103802931E+09	0.1309068094E-28	0.5403805016E+28
107	104:	FE	26	54	0	CO	27	53	0	0.4251641576E+06	0.5042284431E-31	0.5403805016E+28
108	105:	FE	26	54	0	CO	27	54	0	0.8026455518E+07	0.9519069510E-30	0.5403805016E+28
109	106:	FE	26	54	0	CO	27	55	0	0.1175568630E+06	0.1394179471E-31	0.5403805016E+28
110	107:	FE	26	56	0	NN	0	1	0	0.1294835964E+11	0.1052010224E-27	0.7887968323E+29
111	108:	FE	26	56	0	H	1	1	0	0.1322708565E+11	0.1074655765E-27	0.7887968323E+29
112	109:	FE	26	56	0	H	1	2	0	0.6485018658E+09	0.5268857307E-29	0.7887968323E+29
113	110:	FE	26	56	0	H	1	3	0	0.2804930290E+08	0.2278910553E-30	0.7887968323E+29
114	111:	FE	26	56	0	HE	2	3	0	0.1152252073E+09	0.9361656570E-30	0.7887968323E+29
115	112:	FE	26	56	0	HE	2	4	0	0.8396082900E+09	0.6821532069E-29	0.7887968323E+29
116	169:	FE	26	56	0	TI	22	48	0	0.7417367941E+02	0.6026359420E-36	0.7887968323E+29
117	179:	FE	26	56	0	V	23	49	0	0.1002075285E+08	0.8141521197E-31	0.7887968323E+29
118	181:	FE	26	56	0	V	23	51	0	0.3129496674E+07	0.2542609711E-31	0.7887968323E+29
119	188:	FE	26	56	0	CR	24	50	0	0.3047989029E+03	0.2476387520E-35	0.7887968323E+29
120	189:	FE	26	56	0	CR	24	51	0	0.1903672236E+09	0.1546669008E-29	0.7887968323E+29
121	190:	FE	26	56	0	CR	24	52	0	0.2832038545E+09	0.2300935089E-29	0.7887968323E+29
122	191:	FE	26	56	0	CR	24	53	0	0.1158364999E+07	0.9411322022E-32	0.7887968323E+29
123	192:	FE	26	56	0	CR	24	54	0	0.1165756880E+08	0.9471378543E-31	0.7887968323E+29
124	196:	FE	26	56	0	MN	25	51	0	0.9817102039E+01	0.7976061834E-37	0.7887968323E+29
125	197:	FE	26	56	0	MN	25	52	0	0.1545890268E+09	0.1255983315E-29	0.7887968323E+29
126	198:	FE	26	56	0	MN	25	53	0	0.8815593471E+08	0.7162370153E-30	0.7887968323E+29
127	199:	FE	26	56	0	MN	25	54	0	0.1267995071E+10	0.1030202911E-28	0.7887968323E+29
128	200:	FE	26	56	0	MN	25	55	0	0.7253812202E+09	0.5893475939E-29	0.7887968323E+29
129	204:	FE	26	56	0	FE	26	53	0	0.2156782879E+02	0.1752312805E-36	0.7887968323E+29
130	205:	FE	26	56	0	FE	26	54	0	0.2975937641E+10	0.2417848215E-28	0.7887968323E+29
131	206:	FE	26	56	0	FE	26	55	0	0.4948211136E+10	0.4020253414E-28	0.7887968323E+29
132	207:	FE	26	56	0	FE	26	56	0	0.1551234326E+10	0.1260325181E-28	0.7887968323E+29
133	210:	FE	26	56	0	CO	27	55	0	0.3083905589E+09	0.2505568504E-29	0.7887968323E+29
134	211:	FE	26	56	0	CO	27	56	0	0.2971765420E+09	0.2414458427E-29	0.7887968323E+29
135	212:	FE	26	56	0	CO	27	57	0	0.3041404939E+07	0.2471038170E-31	0.7887968323E+29
136	213:	FE	26	57	0	NN	0	1	0	0.3649443293E+09	0.1330164742E-27	0.1758294237E+28
137	214:	FE	26	57	0	H	1	1	0	0.2670335825E+09	0.9732954535E-28	0.1758294237E+28
138	215:	FE	26	57	0	H	1	2	0	0.1672940269E+08	0.6097604437E-29	0.1758294237E+28
139	216:	FE	26	57	0	H	1	3	0	0.1275440383E+07	0.4648779807E-30	0.1758294237E+28
140	217:	FE	26	57	0	HE	2	3	0	0.2836366558E+07	0.1033811047E-29	0.1758294237E+28
141	218:	FE	26	57	0	HE	2	4	0	0.2332170710E+08	0.8500395818E-29	0.1758294237E+28
142	273:	FE	26	57	0	TI	22	49	0	0.7752833017E+00	0.2825785826E-36	0.1758294237E+28
143	282:	FE	26	57	0	V	23	49	0	0.3281858226E+02	0.1196185760E-34	0.1758294237E+28
144	283:	FE	26	57	0	V	23	50	0	0.1835618293E+06	0.6690540276E-31	0.1758294237E+28
145	284:	FE	26	57	0	V	23	51	0	0.6679861824E+00	0.2434704685E-36	0.1758294237E+28
146	285:	FE	26	57	0	V	23	52	0	0.3050010636E+01	0.1111680957E-35	0.1758294237E+28
147	293:	FE	26	57	0	CR	24	51	0	0.3269383508E+04	0.1191638922E-32	0.1758294237E+28
148	294:	FE	26	57	0	CR	24	52	0	0.7230173808E+07	0.2635284756E-29	0.1758294237E+28
149	295:	FE	26	57	0	CR	24	53	0	0.1653080211E+07	0.6025217646E-30	0.1758294237E+28
150	296:	FE	26	57	0	CR	24	54	0	0.4518951495E+05	0.1647086821E-31	0.1758294237E+28
151	297:	FE	26	57	0	CR	24	55	0	0.7219801309E+04	0.2631504144E-32	0.1758294237E+28
152	301:	FE	26	57	0	MN	25	52	0	0.2058316554E+07	0.7502240448E-30	0.1758294237E+28
153	302:	FE	26	57	0	MN	25	53	0	0.9698739328E+07	0.3535038104E-29	0.1758294237E+28
154	303:	FE	26	57	0	MN	25	54	0	0.2412989059E+07	0.8794966005E-30	0.1758294237E+28
155	304:	FE	26	57	0	MN	25	55	0	0.1977898329E+08	0.7209128654E-29	0.1758294237E+28
156	305:	FE	26	57	0	MN	25	56	0	0.5440385638E+07	0.1982935089E-29	0.1758294237E+28
157	309:	FE	26	57	0	FE	26	54	0	0.2296404190E+07	0.8370032473E-30	0.1758294237E+28
158	310:	FE	26	57	0	FE	26	55	0	0.9449535161E+08	0.3444207100E-28	0.1758294237E+28

159 311: FE 26 57 0 FE 26 56 0 0.1023020929E+09 0.3728750553E-28 0.1758294237E+28
 160 312: FE 26 57 0 FE 26 57 0 0.2808424683E+08 0.1023626672E-28 0.1758294237E+28
 161 315: FE 26 57 0 CO 27 55 0 0.4515569494E+07 0.1645854134E-29 0.1758294237E+28
 162 316: FE 26 57 0 CO 27 56 0 0.1114292806E+08 0.4061422208E-29 0.1758294237E+28
 163 317: FE 26 57 0 CO 27 57 0 0.5878105202E+07 0.2142476992E-29 0.1758294237E+28
 164 318: FE 26 57 0 CO 27 58 0 0.3965181061E+05 0.1445246198E-31 0.1758294237E+28
 165 319: FE 26 58 0 NN 0 1 0 0.5793566623E+08 0.1640601429E-27 0.2263149715E+27
 166 320: FE 26 58 0 H 1 1 0 0.2791132972E+08 0.7903830301E-28 0.2263149715E+27
 167 321: FE 26 58 0 H 1 2 0 0.2037087602E+07 0.5768551650E-29 0.2263149715E+27
 168 322: FE 26 58 0 H 1 3 0 0.1593981581E+06 0.4513779906E-30 0.2263149715E+27
 169 323: FE 26 58 0 HE 2 3 0 0.2598326212E+06 0.7357847030E-30 0.2263149715E+27
 170 324: FE 26 58 0 HE 2 4 0 0.2551528049E+07 0.7225325669E-29 0.2263149715E+27
 171 376: FE 26 58 0 TI 22 50 0 0.9741805634E+00 0.2758649600E-35 0.2263149715E+27
 172 385: FE 26 58 0 V 23 50 0 0.2503943210E+00 0.7090576626E-36 0.2263149715E+27
 173 386: FE 26 58 0 V 23 51 0 0.9310532844E+04 0.2636523317E-31 0.2263149715E+27
 174 388: FE 26 58 0 V 23 53 0 0.1852730029E-06 0.5246494484E-42 0.2263149715E+27
 175 396: FE 26 58 0 CR 24 52 0 0.4688542450E+04 0.1327684645E-31 0.2263149715E+27
 176 397: FE 26 58 0 CR 24 53 0 0.1090733133E+06 0.3088698988E-30 0.2263149715E+27
 177 398: FE 26 58 0 CR 24 54 0 0.7904949881E+05 0.2238495372E-30 0.2263149715E+27
 178 399: FE 26 58 0 CR 24 55 0 0.5894353392E-05 0.1669141865E-40 0.2263149715E+27
 179 400: FE 26 58 0 CR 24 56 0 0.3341354293E-04 0.9461927315E-40 0.2263149715E+27
 180 405: FE 26 58 0 MN 25 53 0 0.1200142917E+07 0.3398521693E-29 0.2263149715E+27
 181 406: FE 26 58 0 MN 25 54 0 0.1053914972E+07 0.2984438639E-29 0.2263149715E+27
 182 407: FE 26 58 0 MN 25 55 0 0.2219588486E+06 0.6285351112E-30 0.2263149715E+27
 183 408: FE 26 58 0 MN 25 56 0 0.4947583955E+06 0.1401039090E-29 0.2263149715E+27
 184 409: FE 26 58 0 MN 25 57 0 0.3707982841E+06 0.1050013290E-29 0.2263149715E+27
 185 414: FE 26 58 0 FE 26 55 0 0.4036879251E+06 0.1143148996E-29 0.2263149715E+27
 186 415: FE 26 58 0 FE 26 56 0 0.1504003971E+08 0.4258984535E-28 0.2263149715E+27
 187 416: FE 26 58 0 FE 26 57 0 0.8926880873E+07 0.2527882127E-28 0.2263149715E+27
 188 417: FE 26 58 0 FE 26 58 0 0.3876895164E+07 0.1097845276E-28 0.2263149715E+27
 189 421: FE 26 58 0 CO 27 56 0 0.2685676346E+07 0.7605202007E-29 0.2263149715E+27
 190 422: FE 26 58 0 CO 27 57 0 0.3377108415E+07 0.9563174557E-29 0.2263149715E+27
 191 423: FE 26 58 0 CO 27 58 0 0.7015543919E+06 0.1986636580E-29 0.2263149715E+27
 192 424: FE 26 58 0 CO 27 59 0 0.7735041844E+04 0.2190381423E-31 0.2263149715E+27

193
 194
 195 Prepare full isotope tally 1.2840
 196
 197 -----

198 Full Starting Isotope Tally
 199 Key Element Z A M mk sim Half life Activity Reaction Rate Atoms
 200 -----
 201 000001 NN 000 001 0 000 001 -0.1000000000D+01 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 202 000591 H 001 001 0 000 001 -0.1000000000D+01 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 203 000592 H 001 002 0 000 001 -0.1000000000D+01 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 204 000593 H 001 003 0 000 001 0.3885235200D+09 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 205 001183 HE 002 003 0 000 001 -0.1000000000D+01 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 206 001184 HE 002 004 0 000 001 -0.1000000000D+01 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 207 008881 P 015 031 0 000 001 -0.1000000000D+01 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 208 009472 S 016 032 0 000 001 -0.1000000000D+01 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 209 009473 S 016 033 0 000 001 -0.1000000000D+01 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 210 009474 S 016 034 0 000 001 -0.1000000000D+01 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 211 009476 S 016 036 0 000 001 -0.1000000000D+01 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 212 010062 CL 017 032 0 000 001 0.2980000000D+00 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 213 010063 CL 017 033 0 000 001 0.2511000000D+01 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 214 010064 CL 017 034 0 000 001 0.1526400000D+01 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 215 010065 CL 017 035 0 000 001 -0.1000000000D+01 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 216 010066 CL 017 036 0 000 001 0.9492336000D+13 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 217 010067 CL 017 037 0 000 001 -0.1000000000D+01 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00

218 010068 CL 017 038 0 000 001 0.2234400000D+04 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 219 010069 CL 017 039 0 000 001 0.3372000000D+04 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 220 010070 CL 017 040 0 000 001 0.8100000000D+02 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 221 010071 CL 017 041 0 000 001 0.3840000000D+02 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 222 010653 AR 018 033 0 000 001 0.1730000000D+00 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 223 010654 AR 018 034 0 000 001 0.8445000000D+00 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 224 010655 AR 018 035 0 000 001 0.1775000000D+01 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 225 010656 AR 018 036 0 000 001 -0.1000000000D+01 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 226 010657 AR 018 037 0 000 001 0.3027456000D+07 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 227 010658 AR 018 038 0 000 001 -0.1000000000D+01 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 228 010659 AR 018 039 0 000 001 0.8483184000D+10 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 229 010660 AR 018 040 0 000 001 -0.1000000000D+01 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 230 010661 AR 018 041 0 000 001 0.6576600000D+04 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 231 010662 AR 018 042 0 000 001 0.1037534400D+10 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 232 010663 AR 018 043 0 000 001 0.3222000000D+03 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 233 011245 K 019 035 0 000 001 0.1780000000D+00 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 234 011246 K 019 036 0 000 001 0.3420000000D+00 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 235 011247 K 019 037 0 000 001 0.1226000000D+01 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 236 011248 K 019 038 0 000 001 0.4581600000D+03 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 237 011249 K 019 039 0 000 001 -0.1000000000D+01 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 238 011250 K 019 040 0 000 001 0.3935692800D+17 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 239 011251 K 019 041 0 000 001 -0.1000000000D+01 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 240 011252 K 019 042 0 000 001 0.4449600000D+05 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 241 011253 K 019 043 0 000 001 0.8028000000D+05 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 242 011254 K 019 044 0 000 001 0.1327800000D+04 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 243 011255 K 019 045 0 000 001 0.1038000000D+04 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 244 011256 K 019 046 0 000 001 0.1050000000D+03 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 245 011837 CA 020 037 0 000 001 0.1811000000D+00 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 246 011838 CA 020 038 0 000 001 0.4400000000D+00 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 247 011839 CA 020 039 0 000 001 0.8596000000D+00 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 248 011840 CA 020 040 0 000 001 -0.1000000000D+01 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 249 011841 CA 020 041 0 000 001 0.3216672000D+13 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 250 011842 CA 020 042 0 000 001 -0.1000000000D+01 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 251 011843 CA 020 043 0 000 001 -0.1000000000D+01 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 252 011844 CA 020 044 0 000 001 -0.1000000000D+01 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 253 011845 CA 020 045 0 000 001 0.1404950400D+08 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 254 011846 CA 020 046 0 000 001 -0.1000000000D+01 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 255 011847 CA 020 047 0 000 001 0.3919104000D+06 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 256 011848 CA 020 048 0 000 001 -0.1000000000D+01 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 257 012429 SC 021 039 0 000 001 0.3000000000D-06 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 258 012430 SC 021 040 0 000 001 0.1823000000D+00 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 259 012431 SC 021 041 0 000 001 0.5963000000D+00 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 260 012432 SC 021 042 0 000 001 0.6813000000D+00 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 261 012433 SC 021 043 0 000 001 0.1400760000D+05 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 262 012434 SC 021 044 0 000 001 0.1429200000D+05 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 263 012435 SC 021 045 0 000 001 -0.1000000000D+01 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 264 012436 SC 021 046 0 000 001 0.7239456000D+07 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 265 012437 SC 021 047 0 000 001 0.2893708800D+06 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 266 012438 SC 021 048 0 000 001 0.1572120000D+06 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 267 012439 SC 021 049 0 000 001 0.3432000000D+04 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 268 012440 SC 021 050 0 000 001 0.1025000000D+03 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 269 013021 TI 022 041 0 000 001 0.8040000000D-01 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 270 013022 TI 022 042 0 000 001 0.1990000000D+00 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 271 013023 TI 022 043 0 000 001 0.5090000000D+00 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 272 013024 TI 022 044 0 000 001 0.1892160000D+10 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 273 013025 TI 022 045 0 000 001 0.1108800000D+05 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 274 013026 TI 022 046 0 000 001 -0.1000000000D+01 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 275 013027 TI 022 047 0 000 001 -0.1000000000D+01 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 276 013028 TI 022 048 0 000 001 -0.1000000000D+01 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 277 013029 TI 022 049 0 000 001 -0.1000000000D+01 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 278 013030 TI 022 050 0 000 001 -0.1000000000D+01 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00

279 013031 TI 022 051 0 000 001 0.3456000000D+03 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 280 013032 TI 022 052 0 000 001 0.1020000000D+03 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 281 013613 V 023 043 0 000 001 0.8000000000D+00 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 282 013614 V 023 044 0 000 001 0.1110000000D+00 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 283 013615 V 023 045 0 000 001 0.5470000000D+00 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 284 013616 V 023 046 0 000 001 0.4225000000D+00 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 285 013617 V 023 047 0 000 001 0.1956000000D+04 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 286 013618 V 023 048 0 000 001 0.1380110400D+07 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 287 013619 V 023 049 0 000 001 0.2851200000D+08 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 288 013620 V 023 050 0 000 001 0.4415040000D+25 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 289 013621 V 023 051 0 000 001 -0.1000000000D+01 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 290 013622 V 023 052 0 000 001 0.2245800000D+03 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 291 013623 V 023 053 0 000 001 0.9600000000D+02 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 292 013624 V 023 054 0 000 001 0.4980000000D+02 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 293 013625 V 023 055 0 000 001 0.6540000000D+01 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 294 014205 CR 024 045 0 000 001 0.5000000000D-01 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 295 014206 CR 024 046 0 000 001 0.2600000000D+00 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 296 014207 CR 024 047 0 000 001 0.5000000000D+00 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 297 014208 CR 024 048 0 000 001 0.7776160000D+05 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 298 014209 CR 024 049 0 000 001 0.2538000000D+04 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 299 014210 CR 024 050 0 000 001 -0.1000000000D+01 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 300 014211 CR 024 051 0 000 001 0.2393496000D+07 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 301 014212 CR 024 052 0 000 001 -0.1000000000D+01 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 302 014213 CR 024 053 0 000 001 -0.1000000000D+01 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 303 014214 CR 024 054 0 000 001 -0.1000000000D+01 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 304 014215 CR 024 055 0 000 001 0.2098200000D+03 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 305 014216 CR 024 056 0 000 001 0.3564000000D+03 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 306 014797 MN 025 047 0 000 001 0.1000000000D+00 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 307 014798 MN 025 048 0 000 001 0.1581000000D+00 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 308 014799 MN 025 049 0 000 001 0.3820000000D+00 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 309 014800 MN 025 050 0 000 001 0.2838800000D+00 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 310 014801 MN 025 051 0 000 001 0.2772000000D+04 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 311 014802 MN 025 052 0 000 001 0.4830624000D+06 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 312 014803 MN 025 053 0 000 001 0.1179446400D+15 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 313 014804 MN 025 054 0 000 001 0.2696716800D+08 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 314 014805 MN 025 055 0 000 001 -0.1000000000D+01 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 315 014806 MN 025 056 0 000 001 0.9284040000D+04 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 316 014807 MN 025 057 0 000 001 0.8540000000D+02 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 317 015389 FE 026 049 0 000 001 0.7000000000D-01 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 318 015390 FE 026 050 0 000 001 0.1550000000D+00 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 319 015391 FE 026 051 0 000 001 0.3050000000D+00 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 320 015392 FE 026 052 0 000 001 0.2979000000D+05 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 321 015393 FE 026 053 0 000 001 0.5106000000D+03 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 322 015394 FE 026 054 0 000 001 -0.1000000000D+01 0.0000000000D+00 0.0000000000D+00 0.5405383878D+28
 323 015395 FE 026 055 0 000 001 0.8631403200D+08 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 324 015396 FE 026 056 0 000 001 -0.1000000000D+01 0.0000000000D+00 0.0000000000D+00 0.7890272999D+29
 325 015397 FE 026 057 0 000 001 -0.1000000000D+01 0.0000000000D+00 0.0000000000D+00 0.1758807969D+28
 326 015398 FE 026 058 0 000 001 -0.1000000000D+01 0.0000000000D+00 0.0000000000D+00 0.2263810953D+27
 327 015981 CO 027 051 0 000 001 -0.1000000000D+01 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 328 015982 CO 027 052 0 000 001 0.1150000000D+00 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 329 015983 CO 027 053 0 000 001 0.2400000000D+00 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 330 015984 CO 027 054 0 000 001 0.1932300000D+00 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 331 015985 CO 027 055 0 000 001 0.6310800000D+05 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 332 015986 CO 027 056 0 000 001 0.6672931200D+07 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 333 015987 CO 027 057 0 000 001 0.2347833600D+08 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 334 015988 CO 027 058 0 000 001 0.6122304000D+07 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 335 015989 CO 027 059 0 000 001 -0.1000000000D+01 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 336
 337
 338
 339

```

340
341
342
343
344
345
346
347 =====

348
349 START of SIMULATION
350
351 Simulation time from 0.000000000E+00 to 0.260000000E+06
352 Beam from 0.000000000E+00 to 0.300000000E+03
353
354 =====

355
356
357
358
359 -----
360 Simulation tally at 0.300000000E+03
361 -----
362
363 ----

364 Isotope Tally 1.2880
365 Key Element Z A M mk sim Half life Decay Constant Activity Reaction Rate Start Atoms Beam
      End Atoms Atoms At Time
366 ----

367 000001 NN 000 001 0 000 001 -0.100000000D+01 0.000000000D+00 0.000000000D+00 0.000000000D+00 0.000000000D
      +00 0.4116613889D+13 0.4116613889D+13
368 000593 H 001 003 0 000 001 0.3885235200D+09 0.1784054619D-08 0.1616389319D+02 0.000000000D+00 0.000000000D
      +00 0.9060200854D+10 0.9060200854D+10
369 001183 HE 002 003 0 000 001 -0.100000000D+01 0.000000000D+00 0.000000000D+00 0.000000000D+00 0.000000000D
      +00 0.242400000D+04 0.242400000D+04
370 013027 TI 022 047 0 000 001 -0.100000000D+01 0.000000000D+00 0.000000000D+00 0.000000000D+00 0.000000000D
      +00 0.5316233671D+07 0.5316233671D+07
371 013029 TI 022 049 0 000 001 -0.100000000D+01 0.000000000D+00 0.000000000D+00 0.000000000D+00 0.000000000D
      +00 0.1736864062D+05 0.1736864062D+05
372 013617 V 023 047 0 000 001 0.195600000D+04 0.3543697242D-03 0.3482463420D+05 0.000000000D+00 0.000000000D
      +00 0.9827203574D+08 0.9827203574D+08
373 013619 V 023 049 0 000 001 0.285120000D+08 0.2431071761D-07 0.1160219131D+03 0.000000000D+00 0.000000000D
      +00 0.4772459413D+10 0.4772459413D+10
374 013621 V 023 051 0 000 001 -0.100000000D+01 0.000000000D+00 0.000000000D+00 0.000000000D+00 0.000000000D
      +00 0.2509195042D+07 0.2509195042D+07
375 013622 V 023 052 0 000 001 0.224580000D+03 0.3086415445D-02 0.1841703348D+01 0.000000000D+00 0.000000000D
      +00 0.5967127177D+03 0.5967127177D+03
376 013623 V 023 053 0 000 001 0.960000000D+02 0.7220283131D-02 0.1640359913D-06 0.000000000D+00 0.000000000D
      +00 0.2271877547D-04 0.2271877547D-04
377 014209 CR 024 049 0 000 001 0.253800000D+04 0.2731076362D-03 0.1900894319D+06 0.000000000D+00 0.000000000D
      +00 0.6960238630D+09 0.6960238630D+09
378 014210 CR 024 050 0 000 001 -0.100000000D+01 0.000000000D+00 0.000000000D+00 0.000000000D+00 0.000000000D
      +00 0.4621020723D+09 0.4621020723D+09
379 014211 CR 024 051 0 000 001 0.2393496000D+07 0.2895961307D-06 0.1672958542D+05 0.000000000D+00 0.000000000D
      +00 0.5776867728D+11 0.5776867728D+11
380 014212 CR 024 052 0 000 001 -0.100000000D+01 0.000000000D+00 0.000000000D+00 0.000000000D+00 0.000000000D
      +00 0.1648354473D+08 0.1648354473D+08
381 014213 CR 024 053 0 000 001 -0.100000000D+01 0.000000000D+00 0.000000000D+00 0.000000000D+00 0.000000000D

```

+00 0.4169728000D+07 0.4169728000D+07
 382 014214 CR 024 054 0 000 001 -0.1000000000D+01 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 +00 0.1470632000D+07 0.1470632000D+07
 383 014215 CR 024 055 0 000 001 0.2098200000D+03 0.3303532459D-02 0.4539932896D+04 0.0000000000D+00 0.0000000000D+00
 +00 0.1374266169D+07 0.1374266169D+07
 384 014216 CR 024 056 0 000 001 0.3564000000D+03 0.1944857409D-02 0.1476992152D-04 0.0000000000D+00 0.0000000000D+00
 +00 0.7594346737D-02 0.7594346737D-02
 385 014800 MN 025 050 0 000 001 0.2838800000D+00 0.2441690787D+01 0.1542445948D+07 0.0000000000D+00 0.0000000000D+00
 +00 0.6317122367D+06 0.6317122367D+06
 386 014801 MN 025 051 0 000 001 0.2772000000D+04 0.2500530954D-03 0.1285259866D+06 0.0000000000D+00 0.0000000000D+00
 +00 0.5139947831D+09 0.5139947831D+09
 387 014802 MN 025 052 0 000 001 0.4830624000D+06 0.1434901952D-05 0.1098860016D+06 0.0000000000D+00 0.0000000000D+00
 +00 0.7658084333D+11 0.7658084333D+11
 388 014803 MN 025 053 0 000 001 0.1179446400D+15 0.5876885805D-14 0.6277577829D-03 0.0000000000D+00 0.0000000000D+00
 +00 0.1068181012D+12 0.1068181012D+12
 389 014804 MN 025 054 0 000 001 0.2696716800D+08 0.2570337310D-07 0.9804220660D+04 0.0000000000D+00 0.0000000000D+00
 +00 0.3814371220D+12 0.3814371220D+12
 390 014805 MN 025 055 0 000 001 -0.1000000000D+01 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00
 +00 0.2614130226D+07 0.2614130226D+07
 391 014806 MN 025 056 0 000 001 0.9284040000D+04 0.7466008123D-04 0.1314578074D+06 0.0000000000D+00 0.0000000000D+00
 +00 0.1760750930D+10 0.1760750930D+10
 392 014807 MN 025 057 0 000 001 0.8540000000D+02 0.8116477524D-02 0.3383153467D+06 0.0000000000D+00 0.0000000000D+00
 +00 0.4168253355D+08 0.4168253355D+08
 393 015392 FE 026 052 0 000 001 0.2979000000D+05 0.2326778048D-04 0.7957468329D+05 0.0000000000D+00 0.0000000000D+00
 +00 0.3419951609D+10 0.3419951609D+10
 394 015393 FE 026 053 0 000 001 0.5106000000D+03 0.1357515042D-02 0.6332089595D+08 0.0000000000D+00 0.0000000000D+00
 +00 0.4664471036D+11 0.4664471036D+11
 395 015394 FE 026 054 0 000 001 -0.1000000000D+01 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00 0.5405383878D+28
 +28 0.5405383878D+28 0.5405383878D+28
 396 015395 FE 026 055 0 000 001 0.8631403200D+08 0.8030527187D-08 0.1215087645D+05 0.0000000000D+00 0.0000000000D+00
 +00 0.1513085775D+13 0.1513085775D+13
 397 015396 FE 026 056 0 000 001 -0.1000000000D+01 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00 0.7890272999D+29
 +29 0.7890272999D+29 0.7890272999D+29
 398 015397 FE 026 057 0 000 001 -0.1000000000D+01 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00 0.1758807969D+28
 +28 0.1758807969D+28 0.1758807969D+28
 399 015398 FE 026 058 0 000 001 -0.1000000000D+01 0.0000000000D+00 0.0000000000D+00 0.0000000000D+00 0.2263810953D+27
 +27 0.2263810953D+27 0.2263810953D+27
 400 015983 CO 027 053 0 000 001 0.2400000000D+00 0.2888113252D+01 0.4251641576D+06 0.0000000000D+00 0.0000000000D+00
 +00 0.1472117332D+06 0.1472117332D+06
 401 015984 CO 027 054 0 000 001 0.1932300000D+00 0.3587161313D+01 0.8026455518D+07 0.0000000000D+00 0.0000000000D+00
 +00 0.2237550759D+07 0.2237550759D+07
 402 015985 CO 027 055 0 000 001 0.6310800000D+05 0.1098350733D-04 0.1029731940D+07 0.0000000000D+00 0.0000000000D+00
 +00 0.9375256095D+11 0.9375256095D+11
 403 015986 CO 027 056 0 000 001 0.6672931200D+07 0.1038744683D-06 0.9691497262D+04 0.0000000000D+00 0.0000000000D+00
 +00 0.9330009018D+11 0.9330009018D+11
 404 015987 CO 027 057 0 000 001 0.2347833600D+08 0.2952284100D-07 0.1089088520D+03 0.0000000000D+00 0.0000000000D+00
 +00 0.3688969230D+10 0.3688969230D+10
 405 015988 CO 027 058 0 000 001 0.6122304000D+07 0.1132167205D-06 0.2517465310D+02 0.0000000000D+00 0.0000000000D+00
 +00 0.2223580845D+09 0.2223580845D+09
 406
 407 -----
 408 Simulation tally at 0.2600000000E+06
 409 -----
 410
 411 -----
 412 Isotope Tally 1.7640
 413 Key Element Z A M mk sim Half life Decay Constant Activity Reaction Rate Start Atoms Beam
 End Atoms Atoms At Time
 414 -----

415 000001 NN 000 001 0 000 001 -0.1000000000D+01 0.000000000D+00 0.000000000D+00 0.000000000D+00 0.000000000D
+00 0.4116613889D+13 0.4116613889D+13

416 000593 H 001 003 0 000 001 0.3885235200D+09 0.1784054619D-08 0.1615640588D+02 0.000000000D+00 0.000000000D
+00 0.9060200854D+10 0.9056004063D+10

417 001183 HE 002 003 0 000 001 -0.1000000000D+01 0.000000000D+00 0.000000000D+00 0.000000000D+00 0.000000000D
+00 0.242400000D+04 0.4199214759D+07

418 013027 TI 022 047 0 000 001 -0.1000000000D+01 0.000000000D+00 0.000000000D+00 0.000000000D+00 0.000000000D
+00 0.5316233671D+07 0.1035882694D+09

419 013029 TI 022 049 0 000 001 -0.1000000000D+01 0.000000000D+00 0.000000000D+00 0.000000000D+00 0.000000000D
+00 0.1736364062D+05 0.6440824599D+08

420 013619 V 023 049 0 000 001 0.285120000D+08 0.2431071761D-07 0.1313773643D+03 0.000000000D+00 0.000000000D
+00 0.4772459413D+10 0.5404092399D+10

421 013621 V 023 051 0 000 001 -0.1000000000D+01 0.000000000D+00 0.000000000D+00 0.000000000D+00 0.000000000D
+00 0.2509195042D+07 0.8409826886D+10

422 014210 CR 024 050 0 000 001 -0.1000000000D+01 0.000000000D+00 0.000000000D+00 0.000000000D+00 0.000000000D
+00 0.4621020723D+09 0.4627337845D+09

423 014211 CR 024 051 0 000 001 0.2393496000D+07 0.2895961307D-06 0.1444370965D+05 0.000000000D+00 0.000000000D
+00 0.5776867728D+11 0.4987535437D+11

424 014212 CR 024 052 0 000 001 -0.1000000000D+01 0.000000000D+00 0.000000000D+00 0.000000000D+00 0.000000000D
+00 0.1648354473D+08 0.4857309090D+11

425 014213 CR 024 053 0 000 001 -0.1000000000D+01 0.000000000D+00 0.000000000D+00 0.000000000D+00 0.000000000D
+00 0.4169728000D+07 0.4170359063D+07

426 014214 CR 024 054 0 000 001 -0.1000000000D+01 0.000000000D+00 0.000000000D+00 0.000000000D+00 0.000000000D
+00 0.1470632000D+07 0.2539147607D+10

427 014802 MN 025 052 0 000 001 0.4830624000D+06 0.1434901952D-05 0.4510767081D+05 0.000000000D+00 0.000000000D
+00 0.7658084333D+11 0.3143606485D+11

428 014803 MN 025 053 0 000 001 0.1179446400D+15 0.5876885805D-14 0.1176009917D-02 0.000000000D+00 0.000000000D
+00 0.1068181012D+12 0.2001076685D+12

429 014804 MN 025 054 0 000 001 0.2696716800D+08 0.2570337310D-07 0.9738993802D+04 0.000000000D+00 0.000000000D
+00 0.3814371220D+12 0.3788994450D+12

430 014805 MN 025 055 0 000 001 -0.1000000000D+01 0.000000000D+00 0.000000000D+00 0.000000000D+00 0.000000000D
+00 0.2614130226D+07 0.6439401457D+10

431 015392 FE 026 052 0 000 001 0.297900000D+05 0.2326778048D-04 0.1890116552D+03 0.000000000D+00 0.000000000D
+00 0.3419951609D+10 0.8123321232D+07

432 015394 FE 026 054 0 000 001 -0.1000000000D+01 0.000000000D+00 0.000000000D+00 0.000000000D+00 0.5405383878D
+28 0.5405383878D+28 0.5405383878D+28

433 015395 FE 026 055 0 000 001 0.8631403200D+08 0.8030527187D-08 0.1280863433D+05 0.000000000D+00 0.000000000D
+00 0.1513085775D+13 0.1594992960D+13

434 015396 FE 026 056 0 000 001 -0.1000000000D+01 0.000000000D+00 0.000000000D+00 0.000000000D+00 0.7890272999D
+29 0.7890272999D+29 0.7890272999D+29

435 015397 FE 026 057 0 000 001 -0.1000000000D+01 0.000000000D+00 0.000000000D+00 0.000000000D+00 0.1758807969D
+28 0.1758807969D+28 0.1758807969D+28

436 015398 FE 026 058 0 000 001 -0.1000000000D+01 0.000000000D+00 0.000000000D+00 0.000000000D+00 0.2263810953D
+27 0.2263810953D+27 0.2263810953D+27

437 015985 CO 027 055 0 000 001 0.631080000D+05 0.1098350733D-04 0.5942036078D+05 0.000000000D+00 0.000000000D
+00 0.9375256095D+11 0.5409962319D+10

438 015986 CO 027 056 0 000 001 0.6672931200D+07 0.1038744683D-06 0.9433552328D+04 0.000000000D+00 0.000000000D
+00 0.9330009018D+11 0.9081685308D+11

439 015987 CO 027 057 0 000 001 0.2347833600D+08 0.2952284100D-07 0.1080770318D+03 0.000000000D+00 0.000000000D
+00 0.3688969230D+10 0.3660793752D+10

440 015988 CO 027 058 0 000 001 0.6122304000D+07 0.1132167205D-06 0.2444523420D+02 0.000000000D+00 0.000000000D
+00 0.2223580845D+09 0.2159154063D+09

441

442 -----

443 Gamma Lines at end of Simulation 0.260000000E+06

444 -----

Z	A	M	Gamma Energy/KeV	Gamma count/s
24	51	0	0.3200824000E+03	0.1432815997E+04
25	52	0	0.2005800000E+03	0.3428182982E+02
25	52	0	0.3460199000E+03	0.4420551289E+03
25	52	0	0.3980899000E+03	0.4014582702E+02

450	25	52	0	0.3995700000E+03	0.8254703759E+02
451	25	52	0	0.5020600000E+03	0.9472610871E+02
452	25	52	0	0.6001600000E+03	0.1759199162E+03
453	25	52	0	0.6474699000E+03	0.1804306833E+03
454	25	52	0	0.7442329000E+03	0.4059690373E+05
455	25	52	0	0.8481800000E+03	0.1497574671E+04
456	25	52	0	0.9018899000E+03	0.1984737065E+02
457	25	52	0	0.9355440000E+03	0.4262674892E+05
458	25	52	0	0.1045750000E+04	0.3157536957E+02
459	25	52	0	0.1246278000E+04	0.1899032941E+04
460	25	52	0	0.1247879000E+04	0.1714091491E+03
461	25	52	0	0.1333648000E+04	0.2286958910E+04
462	25	52	0	0.1434092000E+04	0.4510767081E+05
463	25	52	0	0.1645819000E+04	0.2120060528E+02
464	25	52	0	0.1981120000E+04	0.1533660808E+02
465	25	52	0	0.2257419000E+04	0.1217907112E+01
466	25	54	0	0.8348479000E+03	0.9736656444E+04
467	26	52	0	0.1686880000E+03	0.1874526871E+03
468	26	52	0	0.3777479000E+03	0.3106047431E+01
469	26	52	0	0.1039928000E+04	0.1798237987E+00
470	26	55	0	0.1259999000E+03	0.1639505194E-04
471	27	55	0	0.9190000000E+02	0.6907616941E+03
472	27	55	0	0.3853999000E+03	0.3208699482E+03
473	27	55	0	0.4114999000E+03	0.6372833694E+03
474	27	55	0	0.4771999000E+03	0.1198805779E+05
475	27	55	0	0.5199999000E+03	0.4902179764E+03
476	27	55	0	0.8036999000E+03	0.1109675238E+04
477	27	55	0	0.8270000000E+03	0.1247827576E+03
478	27	55	0	0.9311000000E+03	0.4456527059E+05
479	27	55	0	0.9845999000E+03	0.3075003670E+03
480	27	55	0	0.1212799000E+04	0.1559783876E+03
481	27	55	0	0.1316600000E+04	0.4211417476E+04
482	27	55	0	0.1369999000E+04	0.1733589026E+04
483	27	55	0	0.1408499000E+04	0.1002718588E+05
484	27	55	0	0.1555999000E+04	0.2718481506E+02
485	27	55	0	0.1622299000E+04	0.2673916235E+02
486	27	55	0	0.1792100000E+04	0.4857614494E+02
487	27	55	0	0.1940599000E+04	0.8467401411E+01
488	27	55	0	0.2144200000E+04	0.5347832470E+02
489	27	55	0	0.2177600000E+04	0.1738045553E+03
490	27	55	0	0.2578700000E+04	0.2540220423E+02
491	27	55	0	0.2872400000E+04	0.6996747482E+02
492	27	55	0	0.2938900000E+04	0.3386960565E+02
493	27	55	0	0.3108300000E+04	0.3119568941E+01
494	27	56	0	0.2634099000E+03	0.2075380569E+01
495	27	56	0	0.4113799000E+03	0.2358388082E+01
496	27	56	0	0.4865399000E+03	0.5754466920E+01
497	27	56	0	0.6550000000E+03	0.3584749885E+01
498	27	56	0	0.6746999000E+03	0.3584749885E+01
499	27	56	0	0.7335109000E+03	0.1792374942E+02
500	27	56	0	0.7877419000E+03	0.2971568983E+02
501	27	56	0	0.8467709000E+03	0.9427420519E+04
502	27	56	0	0.8527800000E+03	0.4716776164E+01
503	27	56	0	0.8965309000E+03	0.8112855002E+01
504	27	56	0	0.9773729000E+03	0.1366921732E+03
505	27	56	0	0.9973299000E+03	0.1216928250E+02
506	27	56	0	0.1037839000E+04	0.1336733422E+04
507	27	56	0	0.1089030000E+04	0.4716776164E+01
508	27	56	0	0.1140404000E+04	0.1292396669E+02
509	27	56	0	0.1159919000E+04	0.8961874712E+01
510	27	56	0	0.1175101000E+04	0.2158396773E+03

511 27 56 0 0.1198780000E+04 0.4811111687E+01
 512 27 56 0 0.1238281000E+04 0.6311046508E+04
 513 27 56 0 0.1272200000E+04 0.2264052559E+01
 514 27 56 0 0.1335389000E+04 0.1113159175E+02
 515 27 56 0 0.1360215000E+04 0.4046993949E+03
 516 27 56 0 0.1442750000E+04 0.1745207181E+02
 517 27 56 0 0.1462340000E+04 0.6131809013E+01
 518 27 56 0 0.1640404000E+04 0.6792156733E+01
 519 27 56 0 0.1771351000E+04 0.1459370545E+04
 520 27 56 0 0.1810771000E+04 0.6018606385E+02
 521 27 56 0 0.1963714000E+04 0.6829891886E+02
 522 27 56 0 0.2015181000E+04 0.2867799908E+03
 523 27 56 0 0.2034755000E+04 0.7443072787E+03
 524 27 56 0 0.2113123000E+04 0.3547015675E+02
 525 27 56 0 0.2212933000E+04 0.3726253170E+02
 526 27 56 0 0.2276360000E+04 0.1207494698E+02
 527 27 56 0 0.2373700000E+04 0.7735512909E+01
 528 27 56 0 0.2523860000E+04 0.6414815583E+01
 529 27 56 0 0.2598459000E+04 0.1632004553E+04
 530 27 56 0 0.2657400000E+04 0.1981045989E+01
 531 27 56 0 0.3009595000E+04 0.1094292070E+03
 532 27 56 0 0.3201962000E+04 0.3131939373E+03
 533 27 56 0 0.3253415000E+04 0.7660044491E+03
 534 27 56 0 0.3272990000E+04 0.1820674656E+03
 535 27 56 0 0.3369690000E+04 0.1037689813E+01
 536 27 56 0 0.3451152000E+04 0.9169411920E+02
 537 27 56 0 0.3547930000E+04 0.1886710466E+02
 538 27 56 0 0.3600700000E+04 0.1556536134E+01
 539 27 56 0 0.3611799000E+04 0.8018519479E+00
 540 27 57 0 0.1441290000E+02 0.9899856117E+01
 541 27 57 0 0.1220606000E+03 0.9251393926E+02
 542 27 57 0 0.1364735000E+03 0.1154261619E+02
 543 27 57 0 0.3396899000E+03 0.3998850178E-02
 544 27 57 0 0.3523300000E+03 0.3242310955E-02
 545 27 57 0 0.3667999000E+03 0.1296924382E-02
 546 27 57 0 0.5700900000E+03 0.1707617103E-01
 547 27 57 0 0.6924099000E+03 0.1610347774E+00
 548 27 57 0 0.7065399000E+03 0.5403851592E-02
 549 27 58 0 0.8107592000E+03 0.2431078541E+02
 550 27 58 0 0.8639510000E+03 0.1686720915E+00
 551 27 58 0 0.1674725000E+04 0.1271152178E+00
 552
 553 -----
 554 Activities Ordered by Most Active
 555 -----
 556

557 15985 CO 27 55 0 0.5942036078E+05
 558 14802 MN 25 52 0 0.4510767081E+05
 559 14211 CR 24 51 0 0.1444370965E+05
 560 15395 FE 26 55 0 0.1280863433E+05
 561 14804 MN 25 54 0 0.9738993802E+04
 562 15986 CO 27 56 0 0.9433552328E+04
 563 15392 FE 26 52 0 0.1890116552E+03
 564 13619 V 23 49 0 0.1313773643E+03
 565 15987 CO 27 57 0 0.1080770318E+03
 566 15988 CO 27 58 0 0.2444523420E+02
 567 593 H 1 3 0 0.1615640588E+02
 568 14803 MN 25 53 0 0.1176009917E-02
 569
 570 -----

571 Totals 1.8880

572 -----

573
574 Total Activity/Bq: 0.1514219906E+06
575 Total Gamma Power/eV/s: 0.2559334931E+12
576 Total Gamma Power/Watts: 0.4100506435E-07
577 Absorbed Dose*/Grays/s: 0.4078849177E-10
578 Absorbed Dose*/Grays/hr: 0.1468385704E-06
579 Fraction of annual dosage if exposed for 1 hr: 0.1468385634E-03
580
581 Absorbed dose, assumes all energy absorbed, 80kg human, 1m from point-target, 1m surface area exposed to
irradiation.
582
583 Dose Limits:
584 employees 18+ 20 millisieverts/year
585 trainees 18+ 6 millisieverts/year
586 public and under 18s 1 millisieverts/year
587 public and under 18s millisieverts averaged per hour: 0.1140771128E-03
588 Dose averaged over area of skin not exceeding 1cm²
589 Source: <http://www.hse.gov.uk/radiation/ionising/doses/>
590
591
592
593 -----

594 Activity calculation complete. 1.8880
595 -----

Appendix E

Activity V2 Full Results

E.1 36MeV Proton Irradiated Iron

E.1.1 Calculation Settings

Material	Pure Iron
Sheet thickness	0.5mm
Density	7,808 kgm^3
Beam energy	36MeV
Beam projectile	proton
Beam current	0.5 microamps
Beam duration	300s
Simulation end time	260,000s

Table E.1: Simulation settings for 36 MeV proton irradiated iron

E.1.2 Results File

Listing E.1: Isotope Activity at End of Simulation

```

1 13.130 #####
2 13.130 Sim 1
3 13.130 #####
4 13.130 #####
5 13.130 Beam flux: 3120754564499.886
6 13.130 Target depth: 0.0005
7 13.130 Projectile: p
8 13.130 Beam duration/s: 300.0
9 13.130 Sim end time/s: 260000.0
10 13.130
11 13.130 Starting Target Material:
12 13.130
13 13.130 #####
14 13.130 Isotopes
15 13.130 #####
16 13.130 Isotope Percentage by Mass
17 13.130 26 Fe 56 91.753250
18 13.130 26 Fe 54 5.845279
19 13.130 26 Fe 57 2.119074
20 13.130 26 Fe 58 0.282397
21 13.130
22 13.130 Loading exyz /cloud/Code/python/activity/examples/fe/Fe36MeV.exyz
23 13.620 Load complete
24 13.620
25 13.620 Calculate residual reaction rates
26 25.746 Save reaction rates
27 25.746
28 25.746
29 25.746 #####
30 25.746 Particle & Residual Reaction Rates
31 25.746 #####
32 25.746 Source isotope: 54Fe (26054)
33 25.746 helium-3: 8128226.96053868
34 25.746 deuteron: 31267425.752861295
35 25.746 triton: 658063.9037544213
36 25.746 alpha: 47929259.98417321
37 25.746 gamma: 529302786.1997857
38 25.746 proton: 1280534004.9887292

```

39	25.746	neutron:	341994739.706016
40	25.746	52Fe:	10375758.270700272
41	25.746	54Co(meta 1):	1268489.4293415854
42	25.746	52Fe(meta 1):	91996.5249677117
43	25.746	53Fe:	109537376.08734141
44	25.746	53Fe(meta 1):	7878234.293365053
45	25.746	50Mn(meta 1):	447309.355049635
46	25.746	54Co:	6134948.952811248
47	25.746	50Mn:	405358.7041290127
48	25.746	52Mn(meta 1):	51514257.29426319
49	25.746	52Mn:	79833423.71247151
50	25.746	54Fe:	92416418.45220399
51	25.746	55Co:	140959.1816623515
52	25.746	47V:	128374.10459496645
53	25.746	53Mn:	121917363.90080762
54	25.746	49Cr:	2248642.2236076766
55	25.746	53Co:	166474.72453697983
56	25.746	51Mn:	13144593.549430655
57	25.746	51Cr:	2004868.7752378457
58	25.746	52Cr:	39343113.4701649
59	25.746	46Ti:	10113.753559680361
60	25.746	50Cr:	20277932.722341496
61	25.746	49V:	4456332.156319902
62	25.746		
63	25.746		
64	25.746	Source isotope: 56Fe (26056)	
65	25.746	helium-3:	113796332.11233293
66	25.746	deutron:	640048071.8232051
67	25.746	triton:	27462082.512083083
68	25.746	alpha:	784412097.0717504
69	25.746	gamma:	12750097788.074738
70	25.746	proton:	12862634835.97489
71	25.746	neutron:	13361355665.752747
72	25.746	53Fe(meta 2):	35110.96922030936
73	25.746	54Co(meta 1):	2098133.50324677
74	25.746	48Ti:	130153.19806252541
75	25.746	53Fe:	1650122.878507311
76	25.746	48Ti(meta 2):	1.3498721704564602
77	25.747	54Co:	2217228.5579557572
78	25.747	52Mn(meta 1):	51722330.55364108
79	25.747	52Mn:	69812825.68860963
80	25.747	53Mn:	617186326.652764
81	25.747	56Co:	297046346.5343496
82	25.747	57Co:	3332169.8943027295
83	25.747	56Fe:	1324989311.7401316
84	25.747	54Mn:	1105697455.8025718
85	25.747	48V:	6824.981311026873
86	25.747	54Cr:	8886829.249919416
87	25.747	51V:	1908704.2981555585
88	25.747	54Fe:	1768658520.9905515
89	25.747	55Mn:	469054279.81113464
90	25.747	51Mn:	18427975.828979623
91	25.747	51Cr:	157917501.71033847
92	25.747	55Co:	158802991.55014923
93	25.747	55Fe:	3310150310.2897525
94	25.747	52Cr:	198718293.6955533
95	25.747	53Cr:	868047.1046782847
96	25.747	50Cr:	24883.064209825417
97	25.747	49V:	3730773.4294895227
98	25.747		
99	25.747		

100	25.747	Source isotope: 57Fe (26057)
101	25.747	helium-3: 2815186.114818401
102	25.747	deuteron: 16795071.729165718
103	25.747	triton: 1290070.0147642847
104	25.747	alpha: 18996008.688927133
105	25.747	gamma: 318048823.03063726
106	25.747	proton: 263457267.1862257
107	25.747	neutron: 381235458.34554696
108	25.747	58Co(meta 1): 17711.056288183212
109	25.747	58Co: 45119.123356589305
110	25.747	52Mn(meta 1): 756983.0649500752
111	25.747	52Mn: 1130057.9800183107
112	25.747	54Mn: 10749922.705687553
113	25.747	57Co: 6645538.449427711
114	25.747	57Fe: 22282747.27841167
115	25.747	53Mn: 9077540.382290483
116	25.747	56Mn: 5561497.664552677
117	25.747	54Cr: 32475.710066264975
118	25.747	51V: 278.34482695928915
119	25.747	50V: 60535.34442859975
120	25.747	56Co: 7513776.276815263
121	25.747	54Fe: 2335917.272526654
122	25.747	49Ti: 66.6465837718329
123	25.747	55Mn: 17896651.547242593
124	25.747	55Cr: 5259.683969248049
125	25.747	51Cr: 3777.5330594318025
126	25.747	55Co: 1040282.7451955614
127	25.747	55Fe: 52180477.480709895
128	25.747	52Cr: 3736281.5484410194
129	25.747	53Cr: 810918.822403883
130	25.747	52V: 1714.7498473523403
131	25.747	56Fe: 82701492.44833918
132	25.747	49V: 29701.937810047228
133	25.747	
134	25.747	
135	25.747	Source isotope: 58Fe (26058)
136	25.747	helium-3: 265562.0411442721
137	25.747	deuteron: 2046736.073932219
138	25.747	triton: 161109.31102851706
139	25.747	alpha: 2132323.5017096302
140	25.747	gamma: 43596863.655506425
141	25.747	proton: 27781338.358164437
142	25.747	neutron: 60156638.64809996
143	25.747	58Co(meta 1): 393025.0417438573
144	25.747	58Co: 787138.5782124916
145	25.747	58Fe: 3578387.673606792
146	25.747	59Co: 7298.027151484427
147	25.747	55Mn: 917307.4059246042
148	25.747	53Mn: 519923.8215990508
149	25.747	50Ti: 139.09122070067937
150	25.747	56Mn: 495624.1780436759
151	25.747	54Mn: 750596.5413600848
152	25.747	54Cr: 42781.814693838154
153	25.747	51V: 3640.3531430735275
154	25.747	50V: 610.1198082776706
155	25.747	56Co: 528863.6120293873
156	25.747	54Fe: 188.91955540759065
157	25.747	57Co: 2188175.452108935
158	25.747	57Fe: 7076618.451591522
159	25.747	55Cr: 26.068658806358545
160	25.747	57Mn: 347080.97455489024

```

161 25.747 55Co:          3843.437575999017
162 25.747 55Fe:          445703.4711783764
163 25.747 52Cr:          1584.071467022736
164 25.747 53Cr:          94533.22614130769
165 25.747 56Fe:          8672082.613410642
166 25.747
167 25.747
168 25.747 Calculate residual amounts over time
169 25.747
170 25.747 In beam time steps: 101
171 25.747 Out of beam time steps: 201
172 25.747
173 25.748 In beam activity over 101 time steps
174 27.168 In beam activity over 201 time steps
175 30.417 Tally amount by residual isotope
176 30.417 Tally amount of residual isotopes by target isotope
177 30.417 Activity per target isotope by residual radioactive isotope over time
178 30.419 Activity by residual radioactive isotope over time
179 30.420 Activity resulting from each target isotope over time
180 30.422 Total activity from all targets and residuals over time
181 30.424 Activity per target and residual over time
182 30.528 Activity per residual over time
183 30.655 Activity per target over time
184 30.771 Activity data for pie charts
185 30.793 Tally gammas by target -> residual -> energy
186 30.797 Tally gammas by residual -> energy
187 30.800 Tally gammas by target -> energy
188 30.803 Tally gammas -> by gamma energy
189 30.807 Tally gammas by residual and gamma energies
190 30.809 Tally gammas by target and gamma energies
191 30.811 Tally gammas by target, residual and gamma energies
192 30.814 Per target and residual gamma lines over time
193 31.061 Per residual gamma lines over time
194 31.297 Per target gamma lines over time
195 31.542 Tallied gamma lines over time
196 31.790 Per residual gamma lines over time
197 31.966 Per target gamma lines over time
198 32.119 Total activity and energy of gammas over time
199 32.266 Gamma data for pie charts
200 32.294 Activity and Gamma totals over time for charts
201 32.294
202 32.294
203 45.551 Gamma Dose - Beam End
204 45.551 =====
205 45.551 Activity (Bq)      1.1873e+08
206 45.551 Gamma Energy (eV)   9.7722e+13
207 45.551 Gamma Energy (mW)   1.5657e-02
208 45.551 Absorbed Dose (mGy/s) 1.5570e-11
209 45.551 Absorbed Dose (mGy/hr) 5.6051e-08
210 45.551
211 45.552
212 45.552 Gamma Dose - Sim End
213 45.552 =====
214 45.552 Activity (Bq)      1.8420e+05
215 45.552 Gamma Energy (eV)   3.5639e+11
216 45.552 Gamma Energy (mW)   5.7100e-05
217 45.552 Absorbed Dose (mGy/s) 5.6782e-14
218 45.552 Absorbed Dose (mGy/hr) 2.0442e-10
219 45.552
220 45.552
221 45.552 Absorbed Dose Calculations

```

222 45.552 =====
223 45.552 Absorbed dose assumptions:
224 45.552 1. radiation from point, emitted isotropically
225 45.552 2. 80Kg human
226 45.552 3. 1m from point source
227 45.552 4. 1m squared surface area
228 45.552 5. all energy absorbed
229 45.552
230 45.552
231 45.552 Dose Limits
232 45.552 =====
233 45.552 employees 18+ 20 millisieverts/year
234 45.552 trainees 18+ 6 millisieverts/year
235 45.552 public and under 18s 1 millisievert/year
236 45.552 public and under 18s 1.140771128E-04 millisieverts/hour
237 45.552 public and under 18s 3.2E-08 millisieverts/second
238 45.552
239 45.552 Dose averaged over area of skin not exceeding 1cm²
240 45.552 Source: <http://www.hse.gov.uk/radiation/ionising/doses/>
241 45.552
242 45.552 1mW gammas completely absorbed by 80kg human will give public limit for entire year over 80 seconds.
243 45.552
244 45.552 Dose under 1W gammas absorbed for 1s no immediate changes (under 100 Rads)
245 45.552 Dose 1W-2W gammas absorbed for 1s acute radiation syndrome (100-200 Rads)
246 45.552 Dose 10W+ gammas absorbed for 1s mostly fatal (1000+ Rads)

E.1.3 Activity Tallied by Residual Isotope

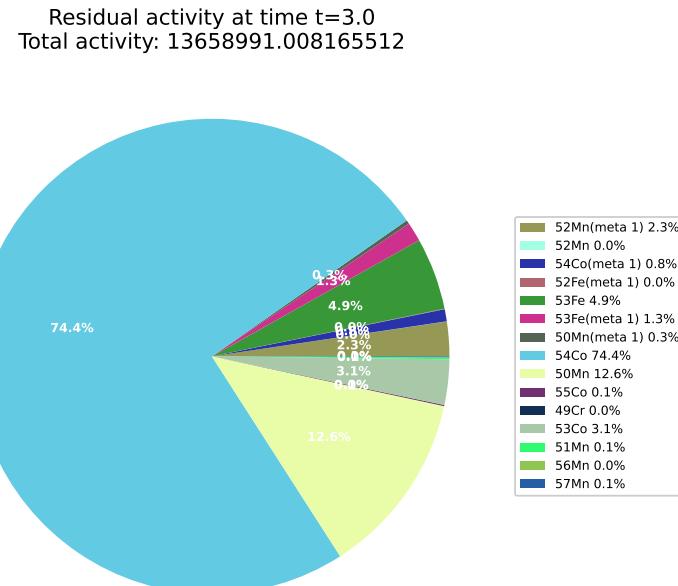


Figure E.1: Residual isotope activity by isotope at 3 seconds into irradiation

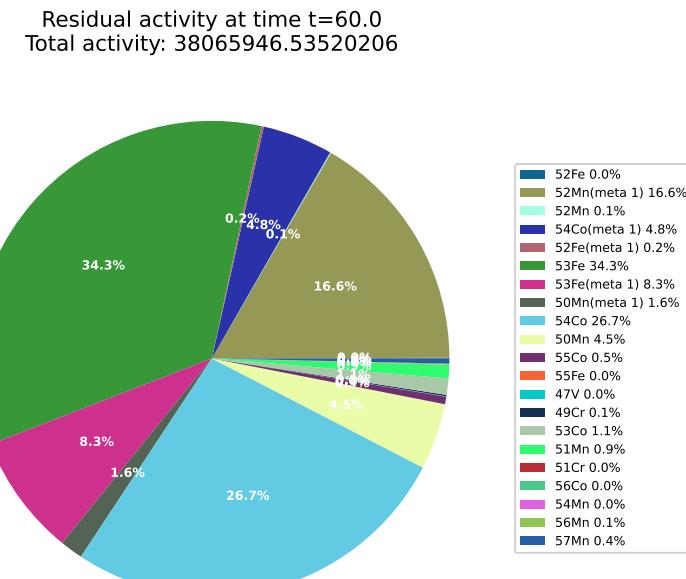


Figure E.2: Residual isotope activity by isotope at 60 seconds into irradiation

Residual activity at time t=300.0
Total activity: 118728909.78631218

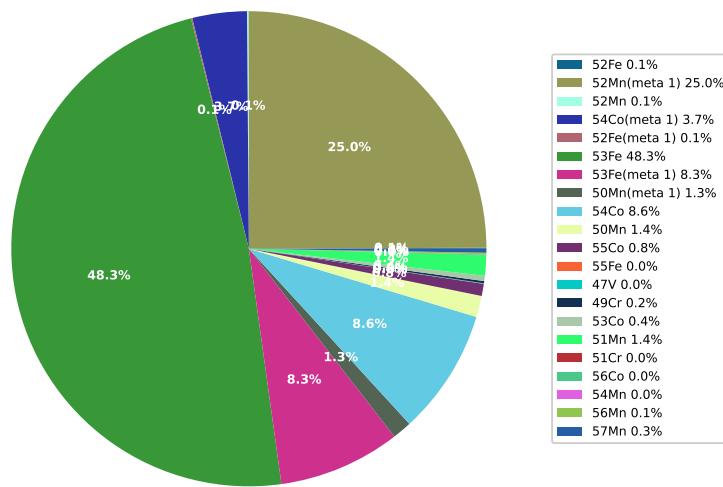


Figure E.3: Residual isotope activity by isotope at 300 seconds into irradiation - end of irradiation

Residual activity at time t=86001.0
Total activity: 554898.1962193253

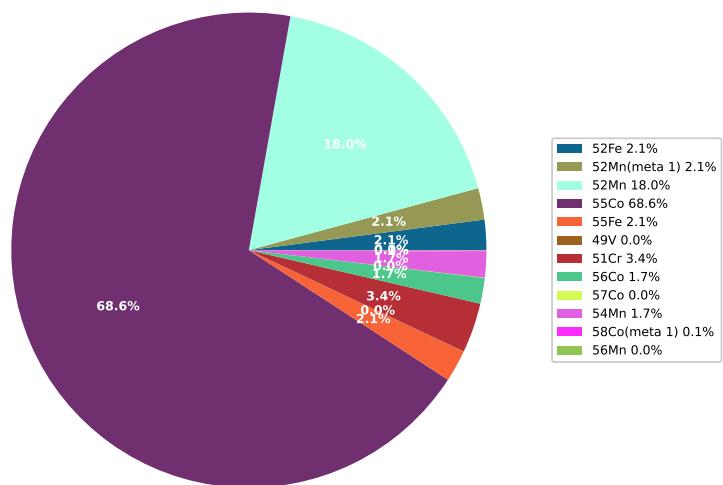


Figure E.4: Residual isotope activity by isotope at 1 day, with approximately 1 day of cooling

Residual activity at time t=173000.5
Total activity: 287692.0694298974

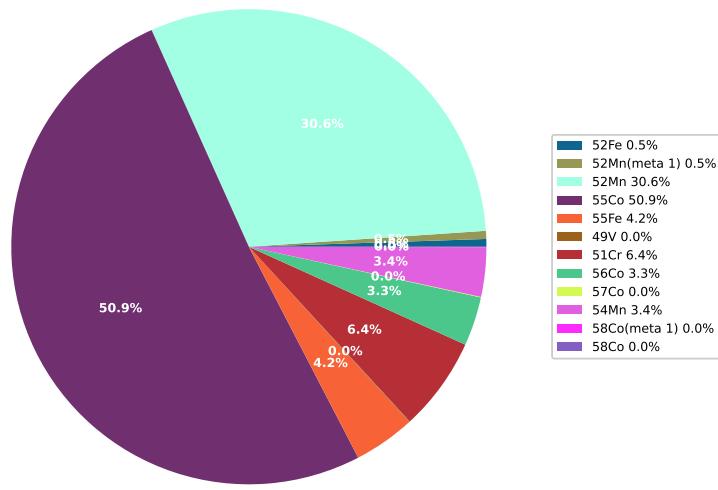


Figure E.5: Residual isotope activity by isotope at 2 days, with approximately 2 days of cooling

Residual activity at time t=260000.0
Total activity: 184007.52249499888

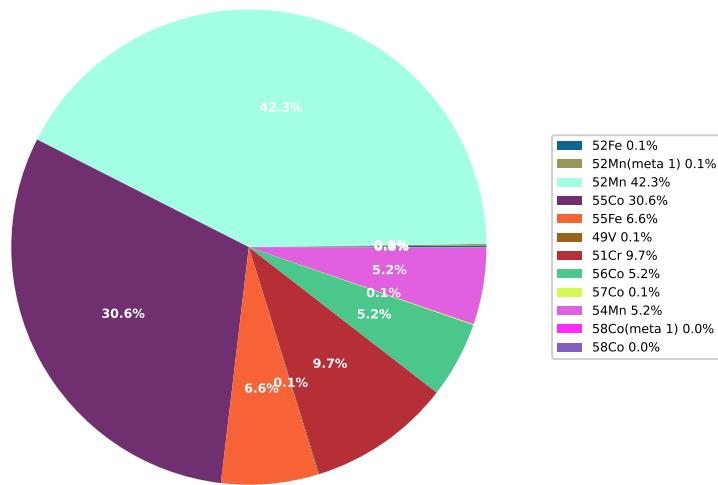


Figure E.6: Residual isotope activity by isotope at approximately 3 days, with approximately 3 days of cooling

E.1.4 Gamma Output Tallied by Residual Isotope

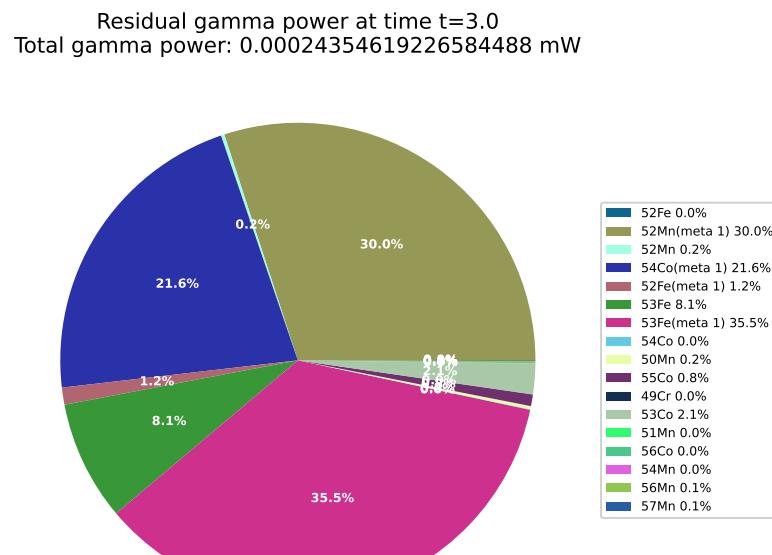


Figure E.7: Gamma activity by isotope at 3 seconds into irradiation

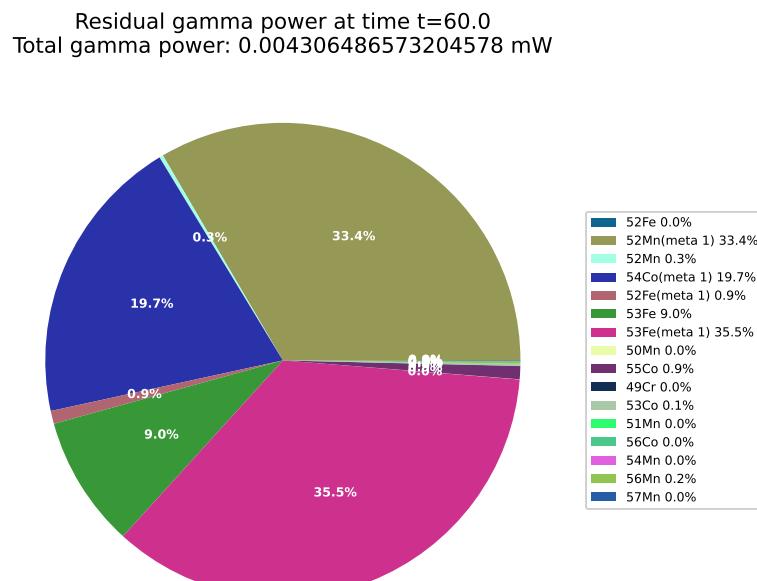


Figure E.8: Gamma activity by isotope at 60 seconds into irradiation

Residual gamma power at time t=300.0
Total gamma power: 0.015656781568754878 mW

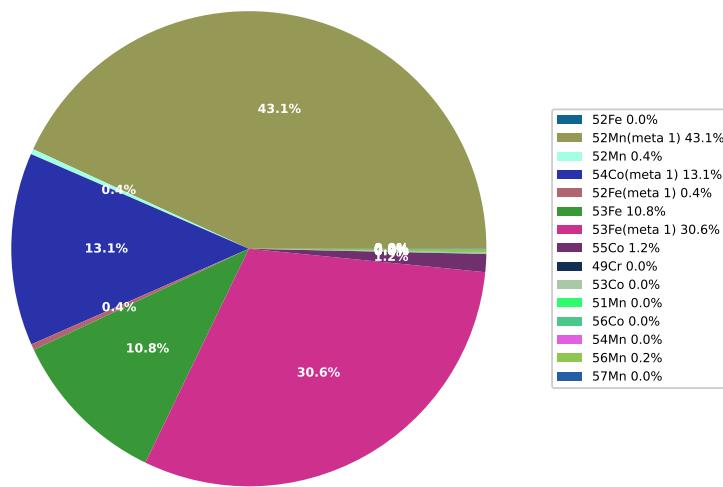


Figure E.9: Gamma activity by isotope at 300 seconds into irradiation

Residual gamma power at time t=86001.0
Total gamma power: 0.00013470195336474222 mW

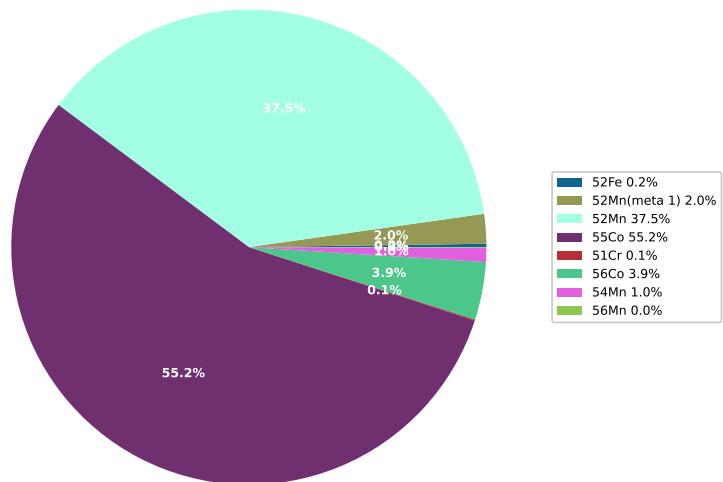


Figure E.10: Gamma activity by isotope with approximately 1 day of cooling

Residual gamma power at time t=173000.5
 Total gamma power: 8.027298909252732e-05 mW

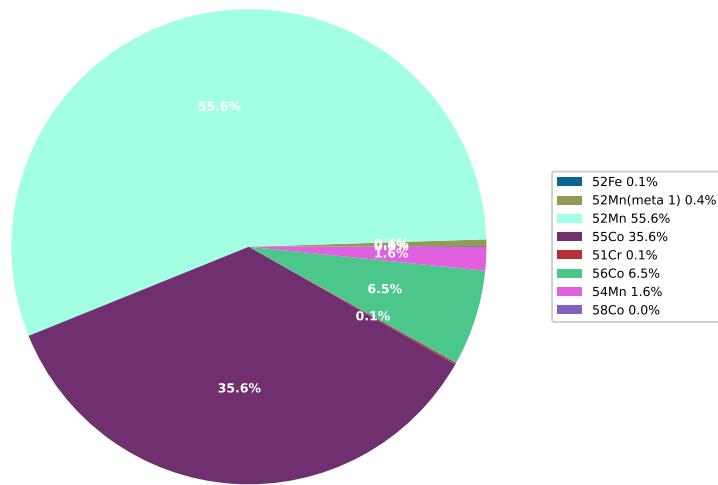


Figure E.11: Gamma activity by isotope with approximately 2 days of cooling

Residual gamma power at time t=260000.0
 Total gamma power: 5.703563656739926e-05 mW

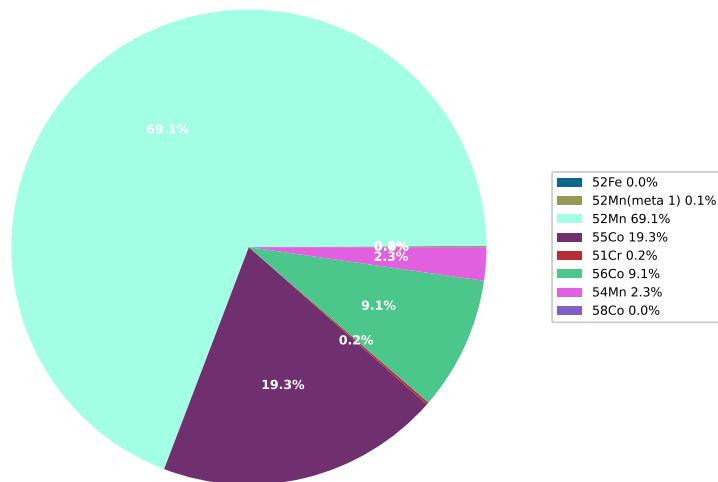


Figure E.12: Gamma activity by isotope with approximately 3 days of cooling

E.1.5 Activity Tallied by Target Isotope

Target activity at time t=3.0
Total activity: 13658991.008165512

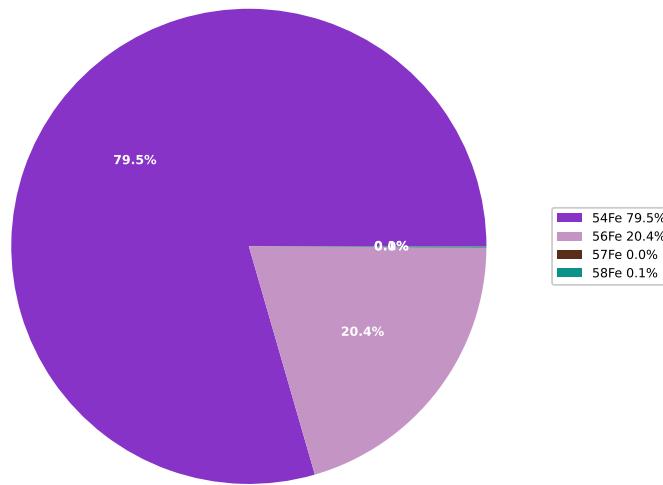


Figure E.13: Isotope activity contribution by target isotope at 3 seconds into irradiation

Target activity at time t=60.0
Total activity: 38065946.53520206

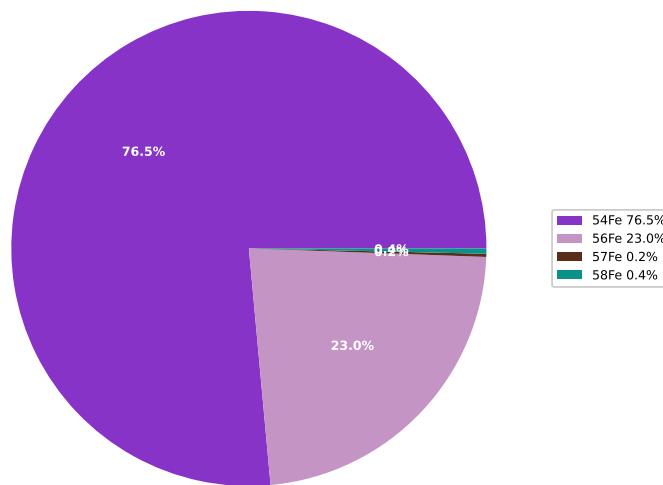


Figure E.14: Isotope activity contribution by target isotope at 60 seconds into irradiation

Target activity at time t=300.0
Total activity: 118728909.78631218

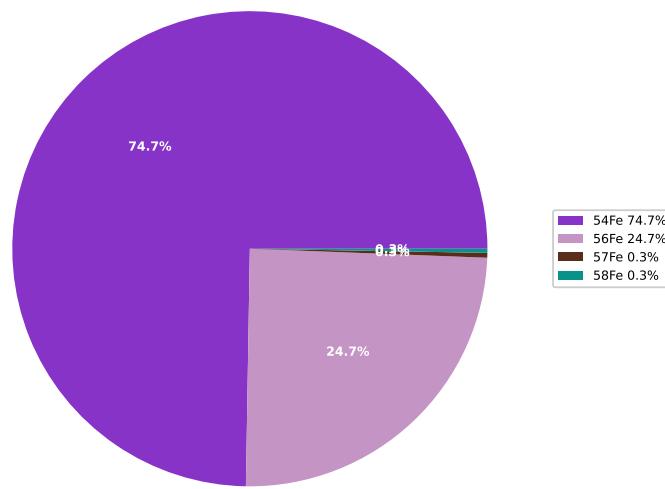


Figure E.15: Isotope activity contribution by target isotope at 300 seconds into irradiation

Target activity at time t=86001.0
Total activity: 554898.1962193253

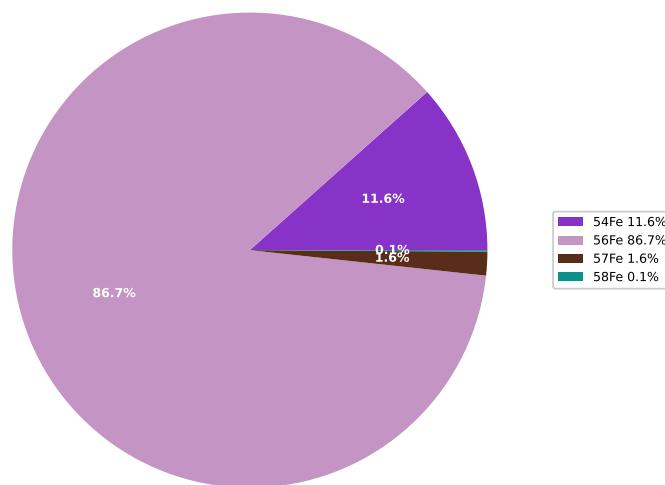


Figure E.16: Isotope activity contribution by target isotope with approximately 1 day of cooling

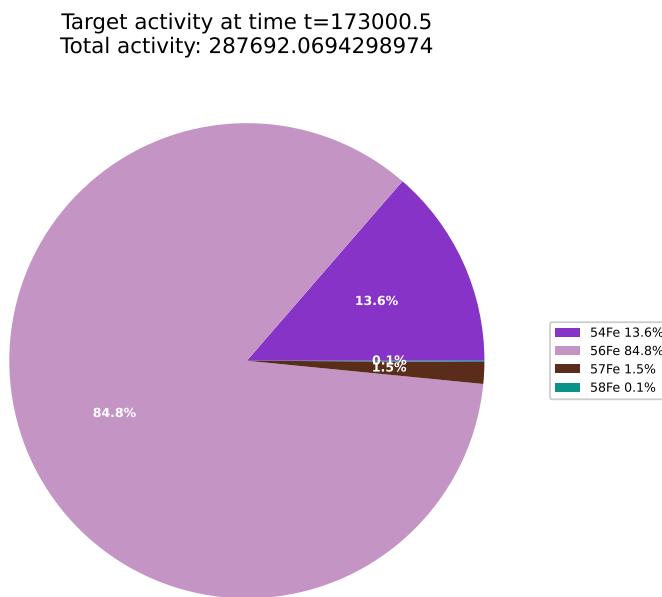


Figure E.17: Isotope activity contribution by target isotope with approximately 2 days of cooling

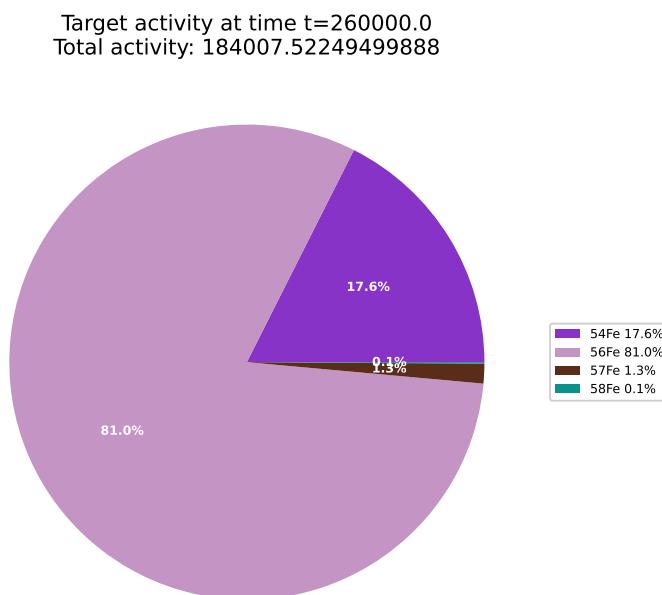


Figure E.18: Isotope activity contribution by target isotope with approximately 3 days of cooling

E.1.6 Gamma Output Tallied by Target Isotope

Target gamma power at time t=3.0
Total gamma power: 0.00024354619226584488 mW

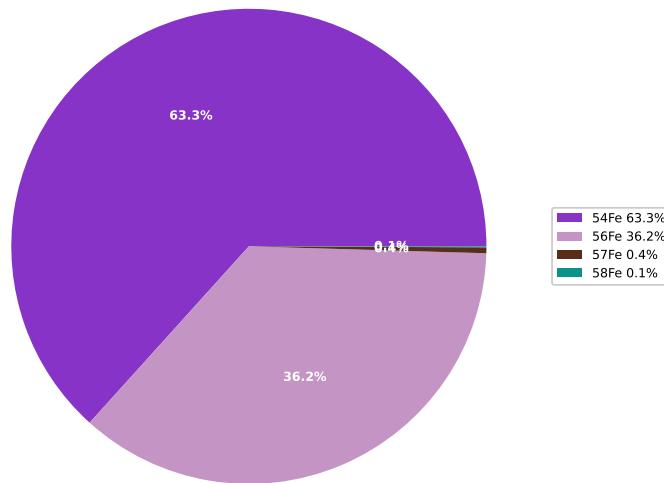


Figure E.19: Gamma activity contribution by target isotope at 3 seconds into irradiation

Target gamma power at time t=60.0
Total gamma power: 0.004306486573204578 mW

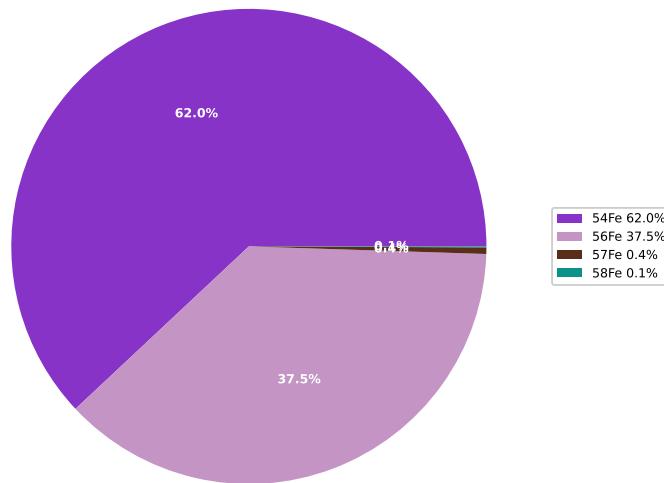


Figure E.20: Gamma activity contribution by target isotope at 60 seconds into irradiation

Target gamma power at time t=300.0
Total gamma power: 0.015656781568754878 mW

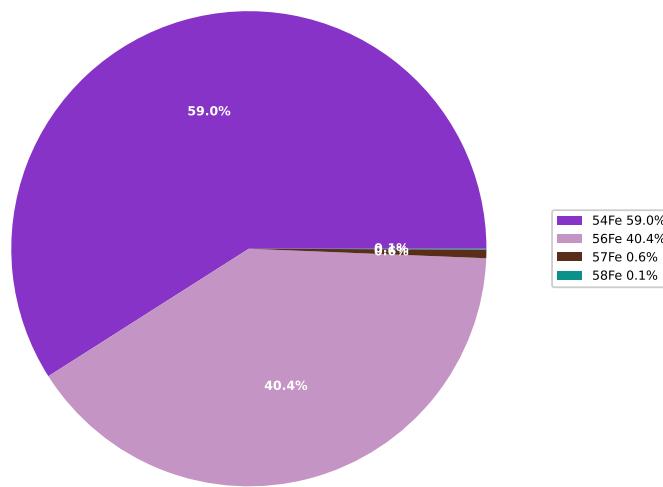


Figure E.21: Gamma activity contribution by target isotope at 300 seconds into irradiation

Target gamma power at time t=86001.0
Total gamma power: 0.00013470195336474222 mW

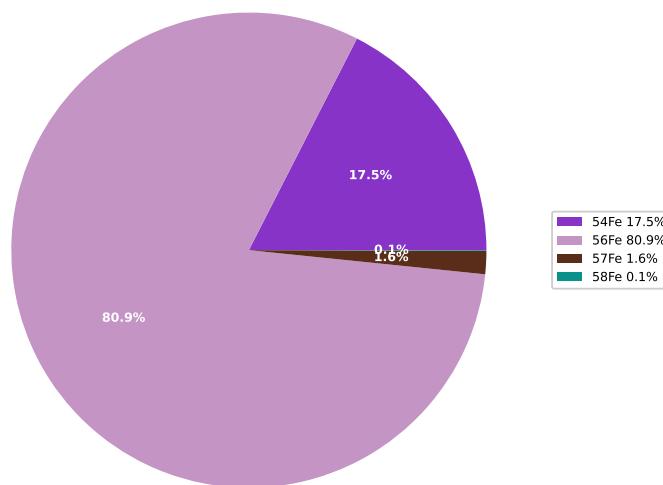


Figure E.22: Gamma activity contribution by target isotope with approximately 1 day of cooling

Target gamma power at time t=173000.5
Total gamma power: 8.027298909252732e-05 mW

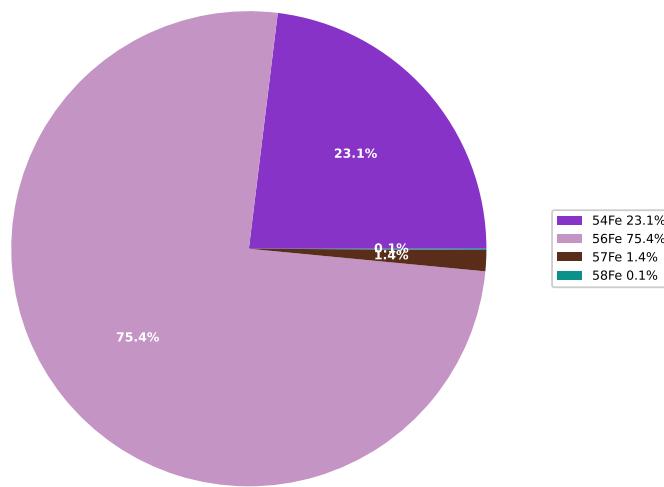


Figure E.23: Gamma activity contribution by target isotope with approximately 2 days of cooling

Target gamma power at time t=260000.0
Total gamma power: 5.703563656739926e-05 mW

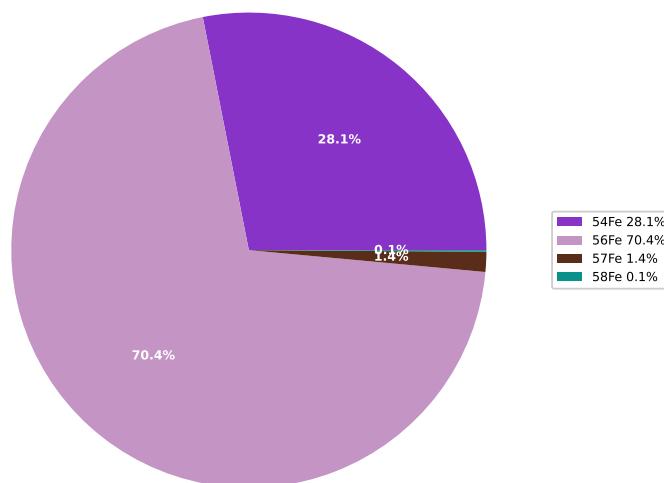


Figure E.24: Gamma activity contribution by target isotope with approximately 3 days of cooling

E.2 13MeV Proton Irradiated Molybdenum

E.2.1 Calculation Settings

Material	Pure Molybdenum
Sheet thickness	506 micrometers
Density	10,220 kgm^3
Beam energy	13MeV
Beam projectile	proton
Beam current	5.0 microamps
Beam duration	1500s
Simulation end time	260,000s

Table E.2: Simulation settings for 13 MeV proton irradiated molybdenum

E.2.2 Results

Listing E.2: Final results for Molybdenum Irradiation

```

1 23.132 #####
2 23.132 Sim 1
3 23.132 #####
4 23.132 #####
5 23.132 Beam flux:      31207545644998.863
6 23.132 Target depth:    0.000506
7 23.132 Projectile:     p
8 23.132 Beam duration/s: 1500.0
9 23.132 Sim end time/s: 606300.0
10 23.132 #####
11 23.132 Starting Target Material:
12 23.132 #####
13 23.132 #####
14 23.132           Isotopes
15 23.132 #####
16 23.132   Isotope      Percentage by Mass
17 23.132   42 Mo 98    24.127937
18 23.132   42 Mo 96    16.678431
19 23.132   42 Mo 95    15.917894
20 23.132   42 Mo 92    14.840324
21 23.132   42 Mo 100   9.630239
22 23.132   42 Mo 97    9.555955
23 23.132   42 Mo 94    9.249221
24 23.132 #####
25 196.630 #####
26 196.630 #####
27 196.630 Particle & Residual Reaction Rates
28 196.630 #####
29 196.630 Source isotope: 92Mo (42092)
30 196.630 #####
31 196.630 #####
32 196.630 Source isotope: 94Mo (42094)
33 196.630 helium-3:      3.1332823136574574e-12
34 196.630 deuteron:     26779.397771044376
35 196.630 triton:       4.691618589935858
36 196.630 alpha:        35497160.708963595
37 196.630 gamma:        4046256131.9846444
38 196.630 proton:       474276321.7635846

```

39 196.630 neutron: 2939219987.127653
 40 196.630 94Tc(meta 1): 982541746.5227002
 41 196.630 95Tc(meta 1): 3220288.2361354767
 42 196.630 95Tc: 4316294.073460804
 43 196.630 91Nb: 16990624.30575451
 44 196.630 91Nb(meta 1): 18484514.956021547
 45 196.630 94Tc: 1914038155.7223403
 46 196.630 93Mo: 42473999.299622655
 47 196.630 93Mo(meta 1): 126.22443599491
 48 196.630 94Mo: 431701273.0135327
 49 196.630 90Zr: 18272.130848077595
 50 196.630
 51 196.630
 52 196.630 Source isotope: 95Mo (42095)
 53 196.630 helium-3: 5.46887072783639e-07
 54 196.630 deuteron: 1234961.9156552393
 55 196.630 triton: 41.10937959624232
 56 196.630 alpha: 54324719.08951578
 57 196.630 gamma: 9102804904.23539
 58 196.630 proton: 894337849.3520026
 59 196.630 neutron: 5518562890.307192
 60 196.630 95Tc(meta 1): 1218252833.6464837
 61 196.630 92Nb(meta 1): 33006155.92287399
 62 196.630 96Tc(meta 1): 2679422.9964109
 63 196.630 95Tc: 3689625934.027424
 64 196.630 92Nb: 14122020.720040252
 65 196.631 96Tc: 4182516.45184412
 66 196.631 94Tc(meta 1): 13905093.155669797
 67 196.631 91Nb: 6611098.919676594
 68 196.631 91Nb(meta 1): 576457.3002812921
 69 196.631 94Tc: 59984103.84506015
 70 196.631 95Mo: 439173380.2083463
 71 196.631 94Mo: 456055572.20441234
 72 196.631
 73 196.631
 74 196.631 Source isotope: 96Mo (42096)
 75 196.631 helium-3: 5.5873148279970936e-15
 76 196.631 deuteron: 89980.87611341325
 77 196.631 triton: 146.48193254666154
 78 196.631 alpha: 16359850.77593771
 79 196.631 gamma: 11880347277.430742
 80 196.631 proton: 319281355.5874101
 81 196.631 neutron: 6458872187.0255995
 82 196.631 93Nb: 10952560.277578255
 83 196.631 97Tc: 5027859.164271742
 84 196.631 96Tc(meta 1): 2314902593.6163425
 85 196.631 97Tc(meta 1): 2657069.708104688
 86 196.631 96Tc: 2988618887.151654
 87 196.631 93Nb(meta 1): 5392146.865275863
 88 196.631 89Y: 4.699677647993188
 89 196.631 89Y(meta 1): 0.024318699111099025
 90 196.631 95Tc(meta 1): 186850567.4939038
 91 196.631 92Nb(meta 1): 2991.5174685434013
 92 196.631 95Tc: 387490658.45324874
 93 196.631 92Nb: 2926.3241012165836
 94 196.631 96Mo: 313532526.5135967
 95 196.631 95Mo: 5619671.213466896
 96 196.631
 97 196.631
 98 196.631 Source isotope: 97Mo (42097)
 99 196.631 helium-3: 2.6147848182753646e-08

```

100 196.631 deutron:      1162205.9105530165
101 196.631 triton:       185.73563697784863
102 196.631 alpha:        10097428.581276963
103 196.631 gamma:        6269889067.613607
104 196.631 proton:       197692275.39840198
105 196.631 neutron:      4202157441.703513
106 196.631 97Tc:         2012907421.2696567
107 196.631 94Nb(meta 1): 7124606.3063902855
108 196.631 97Tc(meta 1): 515939120.74455863
109 196.631 90Y:          0.0007502229001053001
110 196.631 94Nb:         2737836.422765725
111 196.631 90Y(meta 1): 5.501140264263349e-05
112 196.631 93Nb:         173177.71086113452
113 196.631 96Tc(meta 1): 293382016.0592189
114 196.631 96Tc:         530173961.5847317
115 196.631 93Nb(meta 1): 40771.640492079096
116 196.631 98Tc:         2366306.0908124447
117 196.631 97Mo:         172514065.24175954
118 196.631 96Mo:         26186498.548734967
119 196.631
120 196.631
121 196.631 Source isotope: 98Mo (42098)
122 196.631
123 196.631
124 196.631 Source isotope: 100Mo (42100)
125 196.631
126 216.077 Gamma Dose - Beam End
127 216.077 =====
128 216.077 Activity (Bq)      1.2188e+09
129 216.077 Gamma Energy (eV)    7.0916e+14
130 216.077 Gamma Energy (mW)   1.1362e-01
131 216.077 Absorbed Dose (mGy/s) 1.1299e-10
132 216.077 Absorbed Dose (mGy/hr) 4.0676e-07
133 216.077
134 216.077
135 216.077 Gamma Dose - Sim End
136 216.077 =====
137 216.077 Activity (Bq)      6.0485e+06
138 216.077 Gamma Energy (eV)    1.4080e+13
139 216.077 Gamma Energy (mW)   2.2558e-03
140 216.077 Absorbed Dose (mGy/s) 2.2433e-12
141 216.077 Absorbed Dose (mGy/hr) 8.0757e-09
142 216.077
143 216.077
144 216.077 Absorbed Dose Calculations
145 216.077 =====
146 216.077 Absorbed dose assumptions:
147 216.077 1. radiation from point, emitted isotropically
148 216.077 2. 80Kg human
149 216.077 3. 1m from point source
150 216.077 4. 1m squared surface area
151 216.077 5. all energy absorbed
152 216.077
153 216.077
154 216.077 Dose Limits
155 216.077 =====
156 216.077 employees 18+      20 millisieverts/year
157 216.077 trainees 18+      6 millisieverts/year
158 216.077 public and under 18s 1 millisievert/year
159 216.077 public and under 18s 1.140771128E-04 millisieverts/hour
160 216.077 public and under 18s 3.2E-08 millisieverts/second

```

161 216.077
162 216.077 Dose averaged over area of skin not exceeding 1cm²
163 216.077 Source: <http://www.hse.gov.uk/radiation/ionising/doses/>
164 216.077
165 216.077 1mW gammas completely absorbed by 80kg human will give public limit for entire year over 80 seconds.
166 216.077
167 216.077 Dose under 1W gammas absorbed for 1s no immediate changes (under 100 Rads)
168 216.077 Dose 1W-2W gammas absorbed for 1s acute radiation syndrome (100-200 Rads)
169 216.077 Dose 10W+ gammas absorbed for 1s mostly fatal (1000+ Rads)

E.2.3 Total Decay Rates Over Time

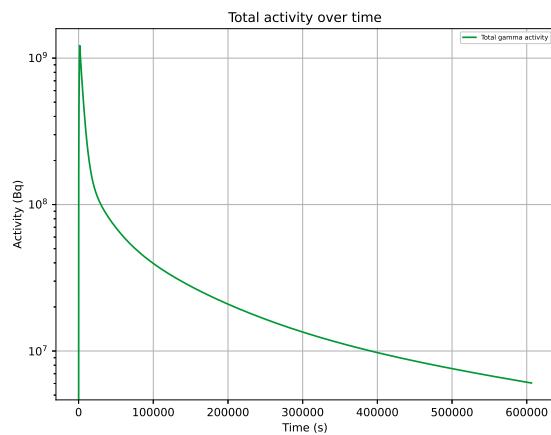


Figure E.25: Total activity over time

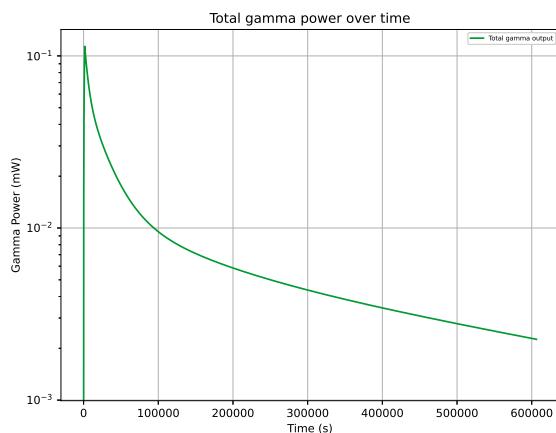


Figure E.26: Total gamma power output over time

E.2.4 Residual Activity Tallied by Target Isotope

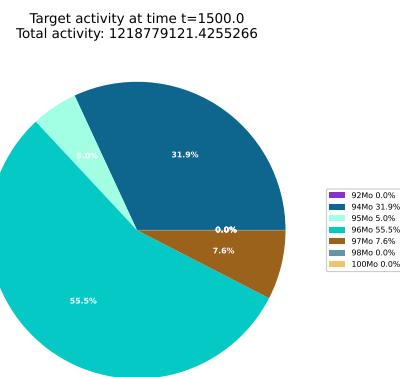


Figure E.27: Radioactive decay from residual isotopes tallied by target isotope - end of beam (1,500 seconds)

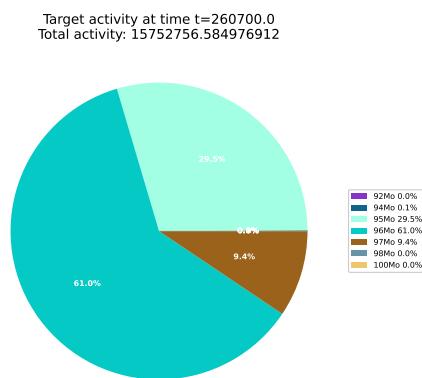


Figure E.28: Radioactive decay from residual isotopes tallied by target isotope - 3 days after irradiation

E.2.5 Residual Activity Tallied by Residual Isotope

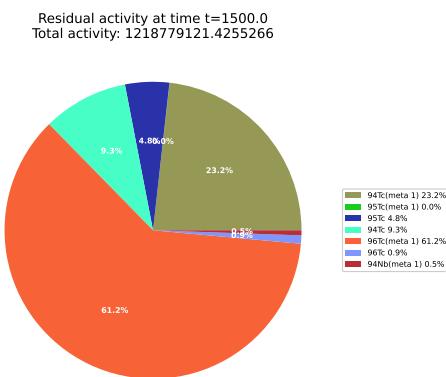


Figure E.29: Radioactive decay from residual isotopes tallied by residual isotope - end of beam (1,500 seconds)

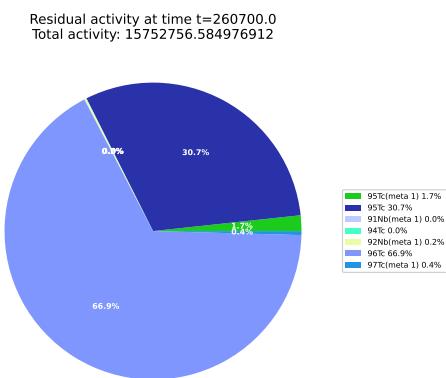


Figure E.30: Radioactive decay from residual isotopes tallied by residual isotope - 3 days after irradiation

E.2.6 Residual Gamma Energy Tallied by Residual Isotope

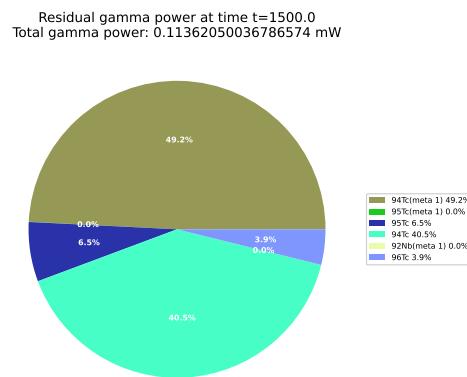


Figure E.31: Radioactive decay from residual isotopes tallied by residual isotope - end of beam (1,500 seconds)

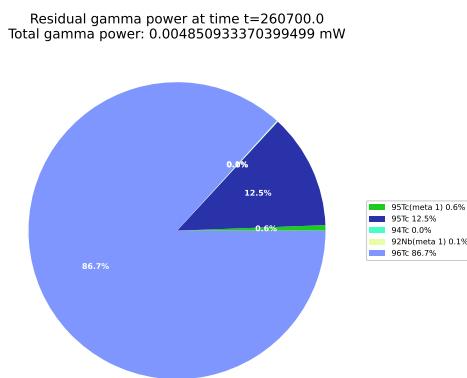


Figure E.32: Radioactive decay from residual isotopes tallied by residual isotope - 3 days after irradiation

E.2.7 Gamma Lines

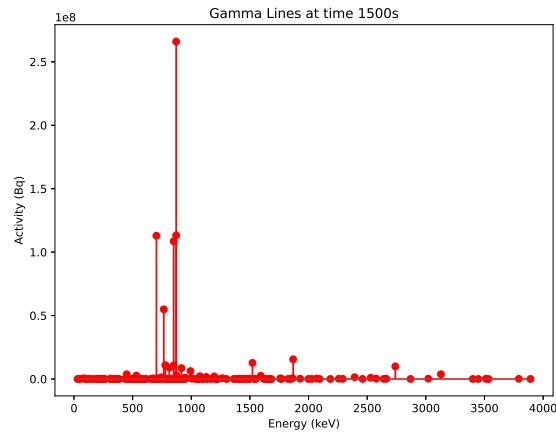


Figure E.33: Predicted gamma lines - end of beam (1,500 seconds)

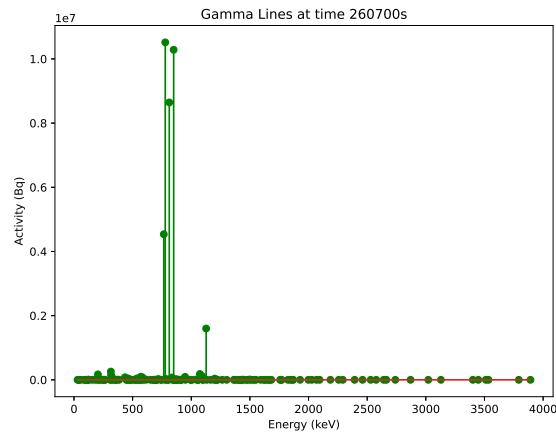


Figure E.34: Predicted gamma lines - 3 days after irradiation

E.3 13MeV Proton Irradiated Steel

E.3.1 Calculation Settings

Material	Steel
Compotision	Fe: 96.375%, C,0.772%, Cu: 0.024%, Mn: 1.36%,
Ni: 0.698%, Si: 0.381%, Cr: 0.092%, V: 0.008%, P:0.009%, Si: 0.003%, Mo: 0.278%	
Sheet thickness	0.1mm
Density	10,220 kgm ³
Beam energy	13MeV
Beam projectile	proton
Beam current	5.0 microamps
Beam duration	1500s
Simulation end time	260,000s

Table E.3: Simulation settings for 13 MeV proton irradiated steel

E.3.2 Results

Listing E.3: Final results for steel irradiation

```

1 22.923 #####
2 22.924 Sim 1 #####
3 22.924 #####
4 22.924 #####
5 22.924 Beam flux: 3120754564499.886
6 22.924 Target depth: 0.0001
7 22.924 Projectile: p
8 22.924 Beam duration/s: 300.0
9 22.924 Sim end time/s: 260000.0
10 22.924 #####
11 22.924 Starting Target Material:
12 22.924 #####
13 22.924 #####
14 22.924 Isotopes #####
15 22.924 #####
16 22.924 Isotope Percentage by Mass
17 22.924 26 Fe 56 88.427066
18 22.924 26 Fe 54 5.633380
19 22.924 26 Fe 57 2.042255
20 22.924 25 Mn 55 1.359982
21 22.924 6 C 12 0.763791
22 22.924 28 Ni 58 0.475171
23 22.924 14 Si 28 0.354158
24 22.924 26 Fe 58 0.272159
25 22.924 28 Ni 60 0.183035
26 22.924 24 Cr 52 0.077085
27 22.924 42 Mo 98 0.067089
28 22.924 42 Mo 96 0.046375
29 22.924 42 Mo 95 0.044261
30 22.924 42 Mo 92 0.041264
31 22.924 42 Mo 100 0.026777
32 22.924 42 Mo 97 0.026571
33 22.924 42 Mo 94 0.025718
34 22.924 28 Ni 62 0.025369
35 22.924 14 Si 29 0.017983

```

```

36 22.924    29 Cu 63    0.016601
37 22.924    14 Si 30    0.011855
38 22.924    15 P 31     0.009000
39 22.924    24 Cr 53    0.008741
40 22.924    6 C 13     0.008322
41 22.924    23 V 51     0.007980
42 22.924    28 Ni 61    0.007957
43 22.924    29 Cu 65    0.007400
44 22.924    28 Ni 64    0.006461
45 22.924    24 Cr 50    0.003997
46 22.924    24 Cr 54    0.002176
47 22.924    23 V 50     0.000020
48 22.924
49 22.924    No neutron data - check input file if this was expected.
50 22.924    Loading exyz /cloud/Code/python/activity/examples/steel_mo_alex/EXYZ.txt
51 25.233    Load complete
52 25.233
53 25.233    Calculate residual reaction rates
54 2093.202  Save reaction rates
55 2093.202
56 2093.202
57 2093.202 #####
58 2093.202 Particle & Residual Reaction Rates
59 2093.202 #####
60 2093.202 Source isotope: 54Fe (26054)
61 2093.202 gamma:        9437.040170446791
62 2093.202 proton:      1565268.3669538412
63 2093.202 54Fe:        1565266.518538786
64 2093.202 55Co:        6775.591316791584
65 2093.202
66 2093.202
67 2093.202 Source isotope: 56Fe (26056)
68 2093.202 alpha:        141.08685329154952
69 2093.202 gamma:        789287.6222288499
70 2093.202 proton:      52520697.287055105
71 2093.202 57Co:        481048.9617216237
72 2093.202 56Fe:        52520681.59831387
73 2093.202 53Mn:        10.002522668635752
74 2093.202
75 2093.202
76 2093.202 Source isotope: 57Fe (26057)
77 2093.202 alpha:        162.32061306305624
78 2093.202 gamma:        535353.864872089
79 2093.202 proton:      978777.8004339848
80 2093.202 neutron:     3688622.628908507
81 2093.202 58Co(meta 1): 728.6876378115435
82 2093.202 58Co:        6881.800814627011
83 2093.202 54Mn:        160.22643404225298
84 2093.202 57Co:        3688627.1514117825
85 2093.202 57Fe:        978777.3418067234
86 2093.202
87 2093.202
88 2093.202 Source isotope: 58Fe (26058)
89 2093.202 alpha:        43.943611364864445
90 2093.202 gamma:        98344.38755516597
91 2093.202 proton:      41017.57430083777
92 2093.202 neutron:     492403.40262528614
93 2093.202 58Co(meta 1): 56789.67542912096
94 2093.202 58Co:        435614.24931542686
95 2093.202 58Fe:        41017.52981489155
96 2093.202 59Co:        1538.890109035841

```

97 2093.202 55Mn: 43.5574480094259
 98 2093.202
 99 2093.202
 100 2093.202 Source isotope: 12C (6012)
 101 2093.202 gamma: 9.297962900075367
 102 2093.202 proton: 0.16469844495501224
 103 2093.202 13N: 0.007674329520104557
 104 2093.202 12C: 0.0015485788201596345
 105 2093.202
 106 2093.202
 107 2093.202 Source isotope: 13C (6013)
 108 2093.202 deuteron: 22007.034476038276
 109 2093.202 alpha: 0.18762061472548627
 110 2093.202 gamma: 2.126677081939579
 111 2093.202 proton: 15575.860814489852
 112 2093.202 neutron: 40043.90782911613
 113 2093.202 14N: 1.5936166964929555
 114 2093.202 13N: 40043.90713723769
 115 2093.202 13C: 15575.759126042365
 116 2093.202 10B: 0.1781070935677874
 117 2093.202 12C: 22007.033370066518
 118 2093.202
 119 2093.202
 120 2093.202 Source isotope: 63Cu (29063)
 121 2093.202 alpha: 1692.0838953018506
 122 2093.202 gamma: 299.2627214435676
 123 2093.202 proton: 6379.155491720199
 124 2093.202 neutron: 5394.987689364512
 125 2093.202 63Zn: 5394.984586472825
 126 2093.202 64Zn: 126.03966676776412
 127 2093.202 60Ni: 1692.0790143093795
 128 2093.202 63Cu: 6379.15167174332
 129 2093.202
 130 2093.203
 131 2093.203 Source isotope: 65Cu (29065)
 132 2093.203 alpha: 179.81444103349426
 133 2093.203 gamma: 4570.928464143155
 134 2093.203 proton: 196.27149662965846
 135 2093.203 neutron: 11626.041286478714
 136 2093.203 65Zn: 11626.108047914477
 137 2093.203 62Ni: 179.8126241955046
 138 2093.203 66Zn: 22.545359039206943
 139 2093.203 65Cu: 196.26876049878186
 140 2093.203
 141 2093.203
 142 2093.203 Source isotope: 55Mn (25055)
 143 2093.203 alpha: 7559.4602195728685
 144 2093.203 gamma: 1089436.2434572722
 145 2093.203 proton: 208787.722901346
 146 2093.203 neutron: 3762868.238725696
 147 2093.203 55Mn: 208787.29464964083
 148 2093.203 55Fe: 3762867.731630595
 149 2093.203 52Cr: 7558.225085207358
 150 2093.203 56Fe: 4020.3882860207686
 151 2093.203
 152 2093.203
 153 2093.203 Source isotope: 58Ni (28058)
 154 2093.203
 155 2093.203
 156 2093.203 Source isotope: 60Ni (28060)
 157 2093.203 alpha: 3.7350612746683702

158 2093.203 gamma: 1068.8890413081508
 159 2093.203 proton: 54079.322487803365
 160 2093.203 61Cu: 700.3202719179782
 161 2093.203 57Co: 3.5767113271911004
 162 2093.203 60Ni: 54079.26781602166
 163 2093.203
 164 2093.203
 165 2093.203 Source isotope: 61Ni (28061)
 166 2093.203 alpha: 1.1816684320989317
 167 2093.203 gamma: 128.82855291693463
 168 2093.203 proton: 5102.959336011498
 169 2093.203 neutron: 7047.509511295016
 170 2093.203 58Co(meta 1): 0.3875470184903323
 171 2093.203 58Co: 0.7908643602588629
 172 2093.203 62Cu: 52.814624350057834
 173 2093.203 61Cu: 7047.508588933512
 174 2093.203 61Ni: 5102.957885289876
 175 2093.203
 176 2093.203
 177 2093.203 Source isotope: 62Ni (28062)
 178 2093.203 alpha: 8.267816891537628
 179 2093.203 gamma: 388.1397871255684
 180 2093.203 proton: 8527.037047829523
 181 2093.203 neutron: 1844.9955288027365
 182 2093.203 62Cu: 1844.9955288027365
 183 2093.203 62Ni: 8527.031845278989
 184 2093.203 59Co: 8.243744437281922
 185 2093.203 63Cu: 208.7277615583037
 186 2093.203
 187 2093.203
 188 2093.203 Source isotope: 64Ni (28064)
 189 2093.203 alpha: 0.07630525559968063
 190 2093.203 gamma: 5151.320921833089
 191 2093.203 proton: 56.99767595200096
 192 2093.203 neutron: 11845.401695288525
 193 2093.203 64Ni: 56.99418413333431
 194 2093.203 64Cu: 11845.473066627274
 195 2093.203 65Cu: 30.98952258201603
 196 2093.203 61Co: 0.06685196871380082
 197 2093.203
 198 2093.203
 199 2093.203 Source isotope: 28Si (14028)
 200 2093.203 gamma: 13.702062307864928
 201 2093.203 proton: 1178590.0836380594
 202 2093.203 28Si: 1178590.0604642944
 203 2093.203 29P: 8.050395593678346
 204 2093.203
 205 2093.203
 206 2093.203 Source isotope: 29Si (14029)
 207 2093.203 gamma: 9.078347487191659
 208 2093.203 proton: 84781.9498117798
 209 2093.203 30P: 7.820937171960048
 210 2093.203 29Si: 84781.94527177457
 211 2093.203
 212 2093.203
 213 2093.203 Source isotope: 30Si (14030)
 214 2093.203 alpha: 18.012799203689656
 215 2093.203 gamma: 44.536475045039495
 216 2093.203 proton: 22280.059950660707
 217 2093.203 27Al: 18.00443071564093
 218 2093.203 30Si: 22280.03863765834

219 2093.203 31P: 37.96948319612531
220 2093.203
221 2093.203
222 2093.203 Source isotope: 50Cr (24050)
223 2093.203
224 2093.203
225 2093.203 Source isotope: 52Cr (24052)
226 2093.203 gamma: 1716.8903887877764
227 2093.203 proton: 44745.238196387676
228 2093.203 53Mn: 1002.4329343842852
229 2093.203 52Cr: 44745.207512644854
230 2093.203
231 2093.203
232 2093.203 Source isotope: 53Cr (24053)
233 2093.203 alpha: 0.0008749097159051281
234 2093.203 gamma: 3819.8654210053855
235 2093.203 proton: 1286.541940389091
236 2093.203 neutron: 24768.975739576643
237 2093.203 53Mn: 24768.990352716894
238 2093.203 54Mn: 30.94645047824251
239 2093.203 53Cr: 1286.5399503518554
240 2093.203
241 2093.203
242 2093.203 Source isotope: 54Cr (24054)
243 2093.203 alpha: 0.4269079942609668
244 2093.203 gamma: 1865.3489179309959
245 2093.203 proton: 221.4928772397376
246 2093.203 neutron: 6388.425420578959
247 2093.203 54Mn: 6388.361753557401
248 2093.203 54Cr: 221.492609742648
249 2093.203 51V: 0.42442091852428715
250 2093.203 55Mn: 7.8571555539379725
251 2093.203
252 2093.203
253 2093.203 Source isotope: 50V (23050)
254 2093.203
255 2093.203
256 2093.203 Source isotope: 51V (23051)
257 2093.203 alpha: 8.793906047440933
258 2093.203 gamma: 2562.072010807901
259 2093.204 proton: 1847.832021588929
260 2093.204 neutron: 25029.202918484196
261 2093.204 48Ti: 8.79377777102894
262 2093.204 48Ti(meta 2): 1.579638927783296e-10
263 2093.204 51V: 1847.8305644822933
264 2093.204 51Cr: 25029.073649384834
265 2093.204 52Cr: 40.93279078441332
266 2093.204
267 2093.204
268 2093.204 Source isotope: 31P (15031)
269 2093.204 alpha: 20500.994041875198
270 2093.204 gamma: 10.0954990307587
271 2093.204 proton: 26364.330767920488
272 2093.204 32S: 9.264165864007683
273 2093.204 28Si: 20500.992316322965
274 2093.204 31P: 26364.329727107408
275 2093.204
276 2093.204
277 2093.204 Source isotope: 92Mo (42092)
278 2093.204
279 2093.204

```

280 2093.204 Source isotope: 94Mo (42094)
281 2093.204 alpha:          6.316988840379401
282 2093.204 gamma:         703.8931041262591
283 2093.204 proton:        415.92814568579826
284 2093.204 95Tc(meta 1): 168.5967957674172
285 2093.204 95Tc:          154.37353711985034
286 2093.204 91Nb:          2.8488760993901363
287 2093.204 91Nb(meta 1): 3.4681060226552836
288 2093.204 94Mo:          415.92255867168075
289 2093.204
290 2093.204
291 2093.204 Source isotope: 95Mo (42095)
292 2093.204 alpha:          0.3612518926731171
293 2093.204 gamma:         2837.9324213752393
294 2093.204 proton:        27.983772297098664
295 2093.204 neutron:       4837.221144492528
296 2093.204 95Tc(meta 1): 1722.2145927476358
297 2093.204 92Nb(meta 1): 0.2650312237060021
298 2093.204 96Tc(meta 1): 10.515615095353965
299 2093.204 95Tc:          3114.998741697279
300 2093.204 92Nb:          0.06524726219560126
301 2093.204 96Tc:          14.174191939937892
302 2093.204 95Mo:          27.966253449261647
303 2093.204
304 2093.204
305 2093.204 Source isotope: 96Mo (42096)
306 2093.204 alpha:          0.3896402613477655
307 2093.204 gamma:         4292.6274939880095
308 2093.204 proton:        21.500556624813015
309 2093.204 neutron:       4752.57364472505
310 2093.204 93Nb:          0.09566857523677805
311 2093.204 97Tc:          132.8134009610271
312 2093.204 96Tc(meta 1): 2590.5566655407283
313 2093.204 97Tc(meta 1): 144.62005700946108
314 2093.204 96Tc:          2162.0149844513953
315 2093.204 93Nb(meta 1): 0.2760343367542316
316 2093.204 96Mo:          21.48595210274134
317 2093.204
318 2093.204
319 2093.204 Source isotope: 97Mo (42097)
320 2093.204 alpha:          0.045466521561900126
321 2093.204 gamma:         3929.5034765150745
322 2093.204 proton:        2.6604200399980016
323 2093.204 neutron:       3129.4062378018352
324 2093.204 deuteron:      2.725811726999687e-37
325 2093.204 97Tc:          2321.9708779633056
326 2093.204 94Nb(meta 1): 0.016376736489314845
327 2093.204 97Tc(meta 1): 807.3747270640994
328 2093.204 94Nb:          0.0032273192037714157
329 2093.204 98Tc:          3.2573253415058523
330 2093.204 97Mo:          2.639546007559529
331 2093.204
332 2093.204
333 2093.204 Source isotope: 98Mo (42098)
334 2093.204
335 2093.204
336 2093.204 Source isotope: 100Mo (42100)
337 2093.204
338 2093.204
339 2093.204 Calculate residual amounts over time
340 2093.204

```

341 2093.204 In beam time steps: 401
 342 2093.204 Out of beam time steps: 201
 343 2093.204
 344 2093.204 In beam activity over 401 time steps
 345 2101.162 Tally amount by residual isotope
 346 2101.163 Tally amount of residual isotopes by target isotope
 347 2101.163 Activity per target isotope by residual radioactive isotope over time
 348 2101.164 Activity by residual radioactive isotope over time
 349 2101.165 Activity resulting from each target isotope over time
 350 2101.167 Total activity from all targets and residuals over time
 351 2101.169 Activity per target and residual over time
 352 2101.381 Activity per residual over time
 353 2101.621 Activity per target over time
 354 2101.858 Activity data for pie charts
 355 2101.947 Tally gammas by target -> residual -> energy
 356 2101.951 Tally gammas by residual -> energy
 357 2101.954 Tally gammas by target -> energy
 358 2101.958 Tally gammas -> by gamma energy
 359 2101.961 Tally gammas by residual and gamma energies
 360 2101.963 Tally gammas by target and gamma energies
 361 2101.965 Tally gammas by target, residual and gamma energies
 362 2101.967 Per target and residual gamma lines over time
 363 2102.366 Per residual gamma lines over time
 364 2102.692 Per target gamma lines over time
 365 2103.016 Tallied gamma lines over time
 366 2103.310 Per residual gamma lines over time
 367 2103.534 Per target gamma lines over time
 368 2103.746 Total activity and energy of gammas over time
 369 2103.937 Gamma full details over time
 370 2106.475 Gamma data for pie charts
 371 2106.589 Activity and Gamma totals over time for charts
 372 2106.589
 373 2106.589
 374
 375 #####
 376 Results
 377 #####
 378
 379
 380 #####
 381 Time Target Residual Activity Half-life/s
 382 #####
 383
 384 0.75s (0.0hr) 54Fe 55Co 0.055814587778344006 63108.0
 385 0.75s (0.0hr) 54Fe 55Fe 1.6736042369180374e-10 86686900.0
 386
 387 0.75s (0.0hr) 56Fe 57Co 0.010649106656255231 23483500.0
 388 0.75s (0.0hr) 56Fe 53Mn 4.477590405477266e-14 116132000000000.0
 389
 390 0.75s (0.0hr) 57Fe 58Co(meta 1) 0.011563273217088194 32760.0
 391 0.75s (0.0hr) 57Fe 58Co 0.0005844341330063274 6121440.0
 392 0.75s (0.0hr) 57Fe 54Mn 3.0880790906605835e-06 26973200.0
 393 0.75s (0.0hr) 57Fe 57Co 0.08165610379858611 23483500.0
 394
 395 0.75s (0.0hr) 58Fe 58Co(meta 1) 0.9011742464423667 32760.0
 396 0.75s (0.0hr) 58Fe 58Co 0.03699436999751045 6121440.0
 397
 398 0.75s (0.0hr) 12C 13N 6.668416321011822e-06 598.02
 399

400	0.75s (0.0hr)	13C	13N	34.79514962336404	598.02
401					
402	0.75s (0.0hr)	63Cu	63Zn	1.2193763813285798	2299.8
403					
404	0.75s (0.0hr)	65Cu	65Zn	0.0002866810350129723	21082500.0
405					
406	0.75s (0.0hr)	55Mn	55Fe	0.022565876380973456	86686900.0
407					
408					
409	0.75s (0.0hr)	60Ni	61Cu	0.030043982065932458	12117.6
410	0.75s (0.0hr)	60Ni	57Co	7.917859393268095e-08	23483500.0
411					
412	0.75s (0.0hr)	61Ni	58Co(meta 1)	6.149839556397991e-06	32760.0
413	0.75s (0.0hr)	61Ni	58Co	6.716403863623248e-08	6121440.0
414	0.75s (0.0hr)	61Ni	62Cu	0.046912880496271896	585.0
415	0.75s (0.0hr)	61Ni	61Cu	0.3023405578072741	12117.6
416					
417	0.75s (0.0hr)	62Ni	62Cu	1.6388274237301088	585.0
418					
419	0.75s (0.0hr)	64Ni	64Cu	0.13468435317630206	45721.4
420	0.75s (0.0hr)	64Ni	64Zn	7.429964442377365e-13	7.25825e+25
421	0.75s (0.0hr)	64Ni	61Co	5.8505335806687024e-06	5940.0
422					
423	0.75s (0.0hr)	28Si	29P	0.9495652750019146	4.142
424					
425	0.75s (0.0hr)	29Si	30P	0.02708001354637464	149.88
426					
427					
428					
429	0.75s (0.0hr)	52Cr	53Mn	4.487352078898796e-12	116132000000000.0
430					
431	0.75s (0.0hr)	53Cr	53Mn	1.1087742285697869e-10	116132000000000.0
432	0.75s (0.0hr)	53Cr	54Mn	5.96437705321599e-07	26973200.0
433					
434	0.75s (0.0hr)	54Cr	54Mn	0.00012312429264658056	26973200.0
435					
436					
437	0.75s (0.0hr)	51V	51Cr	0.005435937241889871	2393630.0
438					
439					
440					
441	0.75s (0.0hr)	94Mo	95Tc(meta 1)	1.6630007396704765e-05	5270400.0
442	0.75s (0.0hr)	94Mo	95Tc	0.001114616624310585	72000.0
443	0.75s (0.0hr)	94Mo	91Nb	6.901505818874096e-11	21459200000.0
444	0.75s (0.0hr)	94Mo	91Nb(meta 1)	3.426478676454852e-07	5261760.0
445					
446	0.75s (0.0hr)	95Mo	95Tc(meta 1)	0.0001698753602388515	5270400.0
447	0.75s (0.0hr)	95Mo	95Tc	0.022491091694841127	72000.0
448	0.75s (0.0hr)	95Mo	92Nb(meta 1)	1.571100087308151e-07	876960.0
449	0.75s (0.0hr)	95Mo	96Tc(meta 1)	0.001768994138678013	3090.0
450	0.75s (0.0hr)	95Mo	96Tc	1.9927543958366358e-05	369792.0
451	0.75s (0.0hr)	95Mo	92Nb	3.070969006681852e-17	1104520000000000.0
452					
453	0.75s (0.0hr)	96Mo	97Tc	8.414941390781586e-13	82049800000000.0
454	0.75s (0.0hr)	96Mo	96Tc(meta 1)	0.43579757491119436	3090.0
455	0.75s (0.0hr)	96Mo	96Tc	0.003039698444859858	369792.0
456	0.75s (0.0hr)	96Mo	97Tc(meta 1)	9.647059513113265e-06	7793280.0
457	0.75s (0.0hr)	96Mo	93Nb(meta 1)	2.820913718831165e-10	508698000.0
458					
459	0.75s (0.0hr)	97Mo	97Tc	1.4711805215270783e-11	82049800000000.0
460	0.75s (0.0hr)	97Mo	94Nb(meta 1)	2.265103167746589e-05	375.6

	Time	Target	Residual	Activity	Half-life/s
461	0.75s (0.0hr)	97Mo	94Nb	2.6703911020600714e-15	630720000000.0
462	0.75s (0.0hr)	97Mo	97Tc(meta 1)	5.385692830187033e-05	7793280.0
463	0.75s (0.0hr)	97Mo	98Tc	1.2479121614454708e-14	135695000000000.0
464					
465					
466					
467					
468					
469	#####	#####	#####	#####	#####
470	Time	Target	Residual	Activity	Half-life/s
471	#####	#####	#####	#####	#####
472					
473	50.25s (0.014hr)	54Fe	55Co	3.738560993240823	63108.0
474	50.25s (0.014hr)	54Fe	55Fe	7.511426371579283e-07	86686900.0
475					
476	50.25s (0.014hr)	56Fe	57Co	0.7134896238985547	23483500.0
477	50.25s (0.014hr)	56Fe	53Mn	2.999985571669768e-12	116132000000000.0
478					
479	50.25s (0.014hr)	57Fe	58Co(meta 1)	0.7743337413479942	32760.0
480	50.25s (0.014hr)	57Fe	58Co	0.03915914760310734	6121440.0
481	50.25s (0.014hr)	57Fe	54Mn	0.00020690116752995448	26973200.0
482	50.25s (0.014hr)	57Fe	57Co	5.4709549513294125	23483500.0
483					
484	50.25s (0.014hr)	58Fe	58Co(meta 1)	60.34706719745709	32760.0
485	50.25s (0.014hr)	58Fe	58Co	2.4787849942804443	6121440.0
486					
487	50.25s (0.014hr)	12C	13N	0.00043421045643969174	598.02
488					
489	50.25s (0.014hr)	13C	13N	2265.6680495850946	598.02
490					
491	50.25s (0.014hr)	63Cu	63Zn	81.09183068156304	2299.8
492					
493	50.25s (0.014hr)	65Cu	65Zn	0.019207613764897446	21082500.0
494					
495	50.25s (0.014hr)	55Mn	55Fe	1.5119134139965489	86686900.0
496					
497					
498	50.25s (0.014hr)	60Ni	61Cu	2.01009969369476	12117.6
499	50.25s (0.014hr)	60Ni	57Co	5.304961911773244e-06	23483500.0
500					
501	50.25s (0.014hr)	61Ni	58Co(meta 1)	0.00041182355402259496	32760.0
502	50.25s (0.014hr)	61Ni	58Co	4.501132308031655e-06	6121440.0
503	50.25s (0.014hr)	61Ni	62Cu	3.052777485942192	585.0
504	50.25s (0.014hr)	61Ni	61Cu	20.228166203341885	12117.6
505					
506	50.25s (0.014hr)	62Ni	62Cu	106.64396237416076	585.0
507					
508	50.25s (0.014hr)	64Ni	64Cu	9.020466624081012	45721.4
509	50.25s (0.014hr)	64Ni	64Zn	7.429964442377365e-13	7.25825e+25
510	50.25s (0.014hr)	64Ni	61Co	0.00039085584374088246	5940.0
511					
512	50.25s (0.014hr)	28Si	29P	8.048601788032434	4.142
513					
514	50.25s (0.014hr)	29Si	30P	1.621774855486964	149.88
515					
516					
517					
518	50.25s (0.014hr)	52Cr	53Mn	3.0065258928621933e-10	116132000000000.0
519					

520	50.25s (0.014hr)	53Cr	53Mn	7.428787331417573e-09	116132000000000.0
521	50.25s (0.014hr)	53Cr	54Mn	3.996130084981988e-05	26973200.0
522					
523	50.25s (0.014hr)	54Cr	54Mn	0.008249322362539599	26973200.0
524					
525					
526	50.25s (0.014hr)	51V	51Cr	0.36420518492744636	2393630.0
527					
528					
529					
530	50.25s (0.014hr)	94Mo	95Tc(meta 1)	0.0011142068677962946	5270400.0
531	50.25s (0.014hr)	94Mo	95Tc	0.07466153318459061	72000.0
532	50.25s (0.014hr)	94Mo	91Nb	4.624057292233371e-09	21459200000.0
533	50.25s (0.014hr)	94Mo	91Nb(meta 1)	2.295733228812532e-05	5261760.0
534					
535	50.25s (0.014hr)	95Mo	95Tc(meta 1)	0.011381612078236538	5270400.0
536	50.25s (0.014hr)	95Mo	95Tc	1.5065442574004724	72000.0
537	50.25s (0.014hr)	95Mo	92Nb(meta 1)	1.052616466794706e-05	876960.0
538	50.25s (0.014hr)	95Mo	96Tc(meta 1)	0.11786702793260355	3090.0
539	50.25s (0.014hr)	95Mo	96Tc	0.0013404518318805023	369792.0
540	50.25s (0.014hr)	95Mo	92Nb	2.0575492344768407e-15	1104520000000000.0
541					
542	50.25s (0.014hr)	96Mo	97Tc	5.638010731823663e-11	82049800000000.0
543	50.25s (0.014hr)	96Mo	96Tc(meta 1)	29.036933369041623	3090.0
544	50.25s (0.014hr)	96Mo	96Tc	0.20497285253117498	369792.0
545	50.25s (0.014hr)	96Mo	97Tc(meta 1)	0.0006463515654757437	7793280.0
546	50.25s (0.014hr)	96Mo	93Nb(meta 1)	1.8900121288447782e-08	508698000.0
547					
548	50.25s (0.014hr)	97Mo	97Tc	9.856909494231425e-10	82049800000000.0
549	50.25s (0.014hr)	97Mo	94Nb(meta 1)	0.0014503810447542796	375.6
550	50.25s (0.014hr)	97Mo	94Nb	2.186889219617751e-13	630720000000.0
551	50.25s (0.014hr)	97Mo	97Tc(meta 1)	0.00360840625812568	7793280.0
552	50.25s (0.014hr)	97Mo	98Tc	8.361011481684656e-13	135695000000000.0
553					
554					
555					
556					
557					
558	#####	#####	#####	#####	#####
559	Time	Target	Residual	Activity	Half-life/s
560	#####	#####	#####	#####	#####
561					
562	100.5s (0.028hr)	54Fe	55Co	7.4750591645609195	63108.0
563	100.5s (0.028hr)	54Fe	55Fe	3.0040173294533986e-06	86686900.0
564					
565	100.5s (0.028hr)	56Fe	57Co	1.4269781895860318	23483500.0
566	100.5s (0.028hr)	56Fe	53Mn	5.999971143339536e-12	116132000000000.0
567					
568	100.5s (0.028hr)	57Fe	58Co(meta 1)	1.5478446435096864	32760.0
569	100.5s (0.028hr)	57Fe	58Co	0.07832247595455813	6121440.0
570	100.5s (0.028hr)	57Fe	54Mn	0.0004138020679218949	26973200.0
571	100.5s (0.028hr)	57Fe	57Co	10.941901788420102	23483500.0
572					
573	100.5s (0.028hr)	58Fe	58Co(meta 1)	120.63000709551221	32760.0
574	100.5s (0.028hr)	58Fe	58Co	4.957899072383595	6121440.0
575					
576	100.5s (0.028hr)	12C	13N	0.0008438534610925341	598.02
577					
578	100.5s (0.028hr)	13C	13N	4403.145518432949	598.02

579						
580	100.5s (0.028hr)	63Cu	63Zn	160.96477280137393	2299.8	
581						
582	100.5s (0.028hr)	65Cu	65Zn	0.038415195798267444	21082500.0	
583						
584	100.5s (0.028hr)	55Mn	55Fe	3.023826220039065	86686900.0	
585						
586						
587	100.5s (0.028hr)	60Ni	61Cu	4.014429883148813	12117.6	
588	100.5s (0.028hr)	60Ni	57Co	1.0609915955500094e-05	23483500.0	
589						
590	100.5s (0.028hr)	61Ni	58Co(meta 1)	0.0008232094872364905	32760.0	
591	100.5s (0.028hr)	61Ni	58Co	9.004581005597246e-06	6121440.0	
592	100.5s (0.028hr)	61Ni	62Cu	5.929099103962157	585.0	
593	100.5s (0.028hr)	61Ni	61Cu	40.39827235570322	12117.6	
594						
595	100.5s (0.028hr)	62Ni	62Cu	207.12371755469115	585.0	
596						
597	100.5s (0.028hr)	64Ni	64Cu	18.03406405706832	45721.4	
598	100.5s (0.028hr)	64Ni	64Zn	7.429964442377365e-13	7.25825e+25	
599	100.5s (0.028hr)	64Ni	61Co	0.0007794265148305085	5940.0	
600						
601	100.5s (0.028hr)	28Si	29P	8.050395193978895	4.142	
602						
603	100.5s (0.028hr)	29Si	30P	2.9072532257752615	149.88	
604						
605						
606						
607	100.5s (0.028hr)	52Cr	53Mn	6.013051785724387e-10	116132000000000.0	
608						
609	100.5s (0.028hr)	53Cr	53Mn	1.4857574662835146e-08	116132000000000.0	
610	100.5s (0.028hr)	53Cr	54Mn	7.992255010407519e-05	26973200.0	
611						
612	100.5s (0.028hr)	54Cr	54Mn	0.01649863407406345	26973200.0	
613						
614						
615	100.5s (0.028hr)	51V	51Cr	0.7284050702016738	2393630.0	
616						
617						
618						
619	100.5s (0.028hr)	94Mo	95Tc(meta 1)	0.0022284063721276894	5270400.0	
620	100.5s (0.028hr)	94Mo	95Tc	0.14928697782598677	72000.0	
621	100.5s (0.028hr)	94Mo	91Nb	9.24815191666299e-09	21459200000.0	
622	100.5s (0.028hr)	94Mo	91Nb(meta 1)	4.5914512609520086e-05	5261760.0	
623						
624	100.5s (0.028hr)	95Mo	95Tc(meta 1)	0.022763148938752323	5270400.0	
625	100.5s (0.028hr)	95Mo	95Tc	3.012360100335684	72000.0	
626	100.5s (0.028hr)	95Mo	92Nb(meta 1)	2.1051911271401087e-05	876960.0	
627	100.5s (0.028hr)	95Mo	96Tc(meta 1)	0.23441291238974704	3090.0	
628	100.5s (0.028hr)	95Mo	96Tc	0.0026915956508928164	369792.0	
629	100.5s (0.028hr)	95Mo	92Nb	4.115098468953681e-15	1104520000000000.0	
630						
631	100.5s (0.028hr)	96Mo	97Tc	1.1276021463647326e-10	82049800000000.0	
632	100.5s (0.028hr)	96Mo	96Tc(meta 1)	57.748398659853486	3090.0	
633	100.5s (0.028hr)	96Mo	96Tc	0.4125915084644436	369792.0	
634	100.5s (0.028hr)	96Mo	97Tc(meta 1)	0.0012927002421992155	7793280.0	
635	100.5s (0.028hr)	96Mo	93Nb(meta 1)	3.780024123177908e-08	508698000.0	
636						
637	100.5s (0.028hr)	97Mo	97Tc	1.971381898846285e-09	82049800000000.0	
638	100.5s (0.028hr)	97Mo	94Nb(meta 1)	0.0027723112729276726	375.6	
639	100.5s (0.028hr)	97Mo	94Nb	5.134863356884003e-13	630720000000.0	

640 100.5s (0.028hr) 97Mo 97Tc(meta 1) 0.007216796389127465 7793280.0
 641 100.5s (0.028hr) 97Mo 98Tc 1.6722022963369312e-12 1356950000000000.0
 642
 643
 644
 645
 646
 647 #####

Time	Target	Residual	Activity	Half-life/s
650				
651 150.0s (0.042hr)	54Fe	55Co	11.153772944664176	63108.0
652 150.0s (0.042hr)	54Fe	55Fe	6.690738505026423e-06	86686900.0
653				
654 150.0s (0.042hr)	56Fe	57Co	2.1298166375882355	23483500.0
655 150.0s (0.042hr)	56Fe	53Mn	8.955180810954531e-12	116132000000000.0
656				
657 150.0s (0.042hr)	57Fe	58Co(meta 1)	2.3090069484237934	32760.0
658 150.0s (0.042hr)	57Fe	58Co	0.1169053600570036	6121440.0
659 150.0s (0.042hr)	57Fe	54Mn	0.0006176146339266021	26973200.0
660 150.0s (0.042hr)	57Fe	57Co	16.33118476925999	23483500.0
661				
662 150.0s (0.042hr)	58Fe	58Co(meta 1)	179.95056916072608	32760.0
663 150.0s (0.042hr)	58Fe	58Co	7.400332844162393	6121440.0
664				
665 150.0s (0.042hr)	12C	13N	0.0012247147993447935	598.02
666				
667 150.0s (0.042hr)	13C	13N	6390.443043406799	598.02
668				
669 150.0s (0.042hr)	63Cu	63Zn	238.4716705046397	2299.8
670				
671 150.0s (0.042hr)	65Cu	65Zn	0.057336066476241516	21082500.0
672				
673 150.0s (0.042hr)	55Mn	55Fe	4.513172569543883	86686900.0
674				
675				
676 150.0s (0.042hr)	60Ni	61Cu	5.9832198284623175	12117.6
677 150.0s (0.042hr)	60Ni	57Co	1.5835683888057496e-05	23483500.0
678				
679 150.0s (0.042hr)	61Ni	58Co(meta 1)	0.0012280279122376598	32760.0
680 150.0s (0.042hr)	61Ni	58Co	1.3443076405121306e-05	6121440.0
681 150.0s (0.042hr)	61Ni	62Cu	8.59989033064392	585.0
682 150.0s (0.042hr)	61Ni	61Cu	60.21072760764531	12117.6
683				
684 150.0s (0.042hr)	62Ni	62Cu	300.42359296293176	585.0
685				
686 150.0s (0.042hr)	64Ni	64Cu	26.906419105746956	45721.4
687 150.0s (0.042hr)	64Ni	64Zn	7.429964442377365e-13	7.25825e+25
688 150.0s (0.042hr)	64Ni	61Co	0.0011599763551937959	5940.0
689				
690 150.0s (0.042hr)	28Si	29P	8.050395593577374	4.142
691				
692 150.0s (0.042hr)	29Si	30P	3.912638144261762	149.88
693				
694				
695				
696 150.0s (0.042hr)	52Cr	53Mn	8.974704157797592e-10	116132000000000.0
697				
698 150.0s (0.042hr)	53Cr	53Mn	2.2175484571395738e-08	116132000000000.0

699	150.0s (0.042hr)	53Cr	54Mn	0.00011928731234452378	26973200.0
700					
701	150.0s (0.042hr)	54Cr	54Mn	0.024624811314053138	26973200.0
702					
703					
704	150.0s (0.042hr)	51V	51Cr	1.0871639547718752	2393630.0
705					
706					
707					
708	150.0s (0.042hr)	94Mo	95Tc(meta 1)	0.003325968833693196	5270400.0
709	150.0s (0.042hr)	94Mo	95Tc	0.22276334223053978	72000.0
710	150.0s (0.042hr)	94Mo	91Nb	1.380326558860933e-08	21459200000.0
711	150.0s (0.042hr)	94Mo	91Nb(meta 1)	6.852889986732321e-05	5261760.0
712					
713	150.0s (0.042hr)	95Mo	95Tc(meta 1)	0.03397473857280299	5270400.0
714	150.0s (0.042hr)	95Mo	95Tc	4.494989234861635	72000.0
715	150.0s (0.042hr)	95Mo	92Nb(meta 1)	3.142014844403659e-05	876960.0
716	150.0s (0.042hr)	95Mo	96Tc(meta 1)	0.34794200031714023	3090.0
717	150.0s (0.042hr)	95Mo	96Tc	0.00403290799680174	369792.0
718	150.0s (0.042hr)	95Mo	92Nb	6.141938013363704e-15	1104520000000000.0
719					
720	150.0s (0.042hr)	96Mo	97Tc	1.6829882781563173e-10	82049800000000.0
721	150.0s (0.042hr)	96Mo	96Tc(meta 1)	85.71666611698106	3090.0
722	150.0s (0.042hr)	96Mo	96Tc	0.6196689367250918	369792.0
723	150.0s (0.042hr)	96Mo	97Tc(meta 1)	0.0019293990993875707	7793280.0
724	150.0s (0.042hr)	96Mo	93Nb(meta 1)	5.641826863745964e-08	508698000.0
725					
726	150.0s (0.042hr)	97Mo	97Tc	2.9423610430541566e-09	82049800000000.0
727	150.0s (0.042hr)	97Mo	94Nb(meta 1)	0.003959992861262792	375.6
728	150.0s (0.042hr)	97Mo	94Nb	8.717414920608875e-13	630720000000.0
729	150.0s (0.042hr)	97Mo	97Tc(meta 1)	0.010771314183369815	7793280.0
730	150.0s (0.042hr)	97Mo	98Tc	2.4958243228909417e-12	1356950000000000.0
731					
732					
733					
734					
735					
736	#####	#####	#####	#####	#####
737	Time	Target	Residual	Activity	Half-life/s
738	#####	#####	#####	#####	#####
739					
740	200.25s (0.056hr)	54Fe	55Co	14.886179632175939	63108.0
741	200.25s (0.056hr)	54Fe	55Fe	1.192220801352644e-05	86686900.0
742					
743	200.25s (0.056hr)	56Fe	57Co	2.8433031026161735	23483500.0
744	200.25s (0.056hr)	56Fe	53Mn	1.19551663826243e-11	116132000000000.0
745					
746	200.25s (0.056hr)	57Fe	58Co(meta 1)	3.0808870431111814	32760.0
747	200.25s (0.056hr)	57Fe	58Co	0.1560769735770539	6121440.0
748	200.25s (0.056hr)	57Fe	54Mn	0.0008245150039385325	26973200.0
749	200.25s (0.056hr)	57Fe	57Co	21.802115498739134	23483500.0
750					
751	200.25s (0.056hr)	58Fe	58Co(meta 1)	240.10641341128093	32760.0
752	200.25s (0.056hr)	58Fe	58Co	9.880099092615463	6121440.0
753					
754	200.25s (0.056hr)	12C	13N	0.0015896313883998717	598.02
755					
756	200.25s (0.056hr)	13C	13N	8294.542413479194	598.02
757					

758	200.25s (0.056hr)	63Cu	63Zn	315.97904157988637	2299.8
759					
760	200.25s (0.056hr)	65Cu	65Zn	0.07654358551646347	21082500.0
761					
762	200.25s (0.056hr)	55Mn	55Fe	6.025084169990238	86686900.0
763					
764					
765	200.25s (0.056hr)	60Ni	61Cu	7.976146139025349	12117.6
766	200.25s (0.056hr)	60Ni	57Co	2.1140622312890297e-05	23483500.0
767					
768	200.25s (0.056hr)	61Ni	58Co(meta 1)	0.00163854651280914	32760.0
769	200.25s (0.056hr)	61Ni	58Co	1.7951115899039187e-05	6121440.0
770	200.25s (0.056hr)	61Ni	62Cu	11.155579142643226	585.0
771	200.25s (0.056hr)	61Ni	61Cu	80.26607350294381	12117.6
772					
773	200.25s (0.056hr)	62Ni	62Cu	389.70254721425994	585.0
774					
775	200.25s (0.056hr)	64Ni	64Cu	35.90639617621384	45721.4
776	200.25s (0.056hr)	64Ni	64Zn	7.429964442377365e-13	7.25825e+25
777	200.25s (0.056hr)	64Ni	61Co	0.001544050296385634	5940.0
778					
779	200.25s (0.056hr)	28Si	29P	8.050395593678322	4.142
780					
781	200.25s (0.056hr)	29Si	30P	4.723075685375102	149.88
782					
783					
784					
785	200.25s (0.056hr)	52Cr	53Mn	1.1981230050659784e-09	116132000000000.0
786					
787	200.25s (0.056hr)	53Cr	53Mn	2.9604271902813308e-08	116132000000000.0
788	200.25s (0.056hr)	53Cr	54Mn	0.00015924845916013463	26973200.0
789					
790	200.25s (0.056hr)	54Cr	54Mn	0.032874101878883	26973200.0
791					
792					
793	200.25s (0.056hr)	51V	51Cr	1.451353320064039	2393630.0
794					
795					
796					
797	200.25s (0.056hr)	94Mo	95Tc(meta 1)	0.004440153721141497	5270400.0
798	200.25s (0.056hr)	94Mo	95Tc	0.29731720012442947	72000.0
799	200.25s (0.056hr)	94Mo	91Nb	1.8427432112335176e-08	21459200000.0
800	200.25s (0.056hr)	94Mo	91Nb(meta 1)	9.148577852438231e-05	5261760.0
801					
802	200.25s (0.056hr)	95Mo	95Tc(meta 1)	0.04535612612200328	5270400.0
803	200.25s (0.056hr)	95Mo	95Tc	5.999360164324543	72000.0
804	200.25s (0.056hr)	95Mo	92Nb(meta 1)	4.194506520753931e-05	876960.0
805	200.25s (0.056hr)	95Mo	96Tc(meta 1)	0.4619090291438015	3090.0
806	200.25s (0.056hr)	95Mo	96Tc	0.005404915292026131	369792.0
807	200.25s (0.056hr)	95Mo	92Nb	8.199487247840544e-15	1104520000000000.0
808					
809	200.25s (0.056hr)	96Mo	97Tc	2.2467893513386836e-10	82049800000000.0
810	200.25s (0.056hr)	96Mo	96Tc(meta 1)	113.7928217675642	3090.0
811	200.25s (0.056hr)	96Mo	96Tc	0.8324507936749032	369792.0
812	200.25s (0.056hr)	96Mo	97Tc(meta 1)	0.0025757420417801347	7793280.0
813	200.25s (0.056hr)	96Mo	93Nb(meta 1)	7.531838602506962e-08	508698000.0
814					
815	200.25s (0.056hr)	97Mo	97Tc	3.928051992477299e-09	82049800000000.0
816	200.25s (0.056hr)	97Mo	94Nb(meta 1)	0.005059663092108257	375.6
817	200.25s (0.056hr)	97Mo	94Nb	1.2982374959935042e-12	630720000000.0
818	200.25s (0.056hr)	97Mo	97Tc(meta 1)	0.014379672301150559	7793280.0

819	200.25s (0.056hr)	97Mo	98Tc	3.3319254710594074e-12	135695000000000.0
820					
821					
822					
823					
824					
825	#####	#####	#####	#####	#####
826	Time	Target	Residual	Activity	Half-life/s
827	#####	#####	#####	#####	#####
828					
829	249.75s (0.069hr)	54Fe	55Co	18.560865202663734	63108.0
830	249.75s (0.069hr)	54Fe	55Fe	1.854145921090801e-05	86686900.0
831					
832	249.75s (0.069hr)	56Fe	57Co	3.546139481166703	23483500.0
833	249.75s (0.069hr)	56Fe	53Mn	1.4910376050239297e-11	116132000000000.0
834					
835	249.75s (0.069hr)	57Fe	58Co(meta 1)	3.840444575329131	32760.0
836	249.75s (0.069hr)	57Fe	58Co	0.19466801007425735	6121440.0
837	249.75s (0.069hr)	57Fe	54Mn	0.0010283270475086531	26973200.0
838	249.75s (0.069hr)	57Fe	57Co	27.191382611265727	23483500.0
839					
840	249.75s (0.069hr)	58Fe	58Co(meta 1)	299.3019088281487	32760.0
841	249.75s (0.069hr)	58Fe	58Co	12.323174590494997	6121440.0
842					
843	249.75s (0.069hr)	12C	13N	0.0019289088052710573	598.02
844					
845	249.75s (0.069hr)	13C	13N	10064.859069723912	598.02
846					
847	249.75s (0.069hr)	63Cu	63Zn	391.19044260937596	2299.8
848					
849	249.75s (0.069hr)	65Cu	65Zn	0.09546439414110316	21082500.0
850					
851	249.75s (0.069hr)	55Mn	55Fe	7.514429331832644	86686900.0
852					
853					
854	249.75s (0.069hr)	60Ni	61Cu	9.933734415962824	12117.6
855	249.75s (0.069hr)	60Ni	57Co	2.6366374858591404e-05	23483500.0
856					
857	249.75s (0.069hr)	61Ni	58Co(meta 1)	0.0020425114515680858	32760.0
858	249.75s (0.069hr)	61Ni	58Co	2.2394128727353758e-05	6121440.0
859	249.75s (0.069hr)	61Ni	62Cu	13.528648695941389	585.0
860	249.75s (0.069hr)	61Ni	61Cu	99.9658033958523	12117.6
861					
862	249.75s (0.069hr)	62Ni	62Cu	472.60198594459	585.0
863					
864	249.75s (0.069hr)	64Ni	64Cu	44.7653442924128	45721.4
865	249.75s (0.069hr)	64Ni	64Zn	7.429964442377365e-13	7.25825e+25
866	249.75s (0.069hr)	64Ni	61Co	0.0019201962278281332	5940.0
867					
868	249.75s (0.069hr)	28Si	29P	8.050395593678346	4.142
869					
870	249.75s (0.069hr)	29Si	30P	5.356926631484682	149.88
871					
872					
873					
874	249.75s (0.069hr)	52Cr	53Mn	1.494288242273299e-09	116132000000000.0
875					
876	249.75s (0.069hr)	53Cr	53Mn	3.6922181811373905e-08	116132000000000.0
877	249.75s (0.069hr)	53Cr	54Mn	0.0001986131204965335	26973200.0

878
 879 249.75s (0.069hr) 54Cr 54Mn 0.04100025828896936 26973200.0
 880
 881
 882 249.75s (0.069hr) 51V 51Cr 1.8101018418226418 2393630.0
 883
 884
 885
 886 249.75s (0.069hr) 94Mo 95Tc(meta 1) 0.005537701784079425 5270400.0
 887 249.75s (0.069hr) 94Mo 95Tc 0.3707230800913098 72000.0
 888 249.75s (0.069hr) 94Mo 91Nb 2.2982616761795117e-08 21459200000.0
 889 249.75s (0.069hr) 94Mo 91Nb(meta 1) 0.00011409986862291262 5261760.0
 890
 891 249.75s (0.069hr) 95Mo 95Tc(meta 1) 0.05656756867421637 5270400.0
 892 249.75s (0.069hr) 95Mo 95Tc 7.480566634503592 72000.0
 893 249.75s (0.069hr) 95Mo 92Nb(meta 1) 5.231248495802371e-05 876960.0
 894 249.75s (0.069hr) 95Mo 96Tc(meta 1) 0.5729260152583443 3090.0
 895 249.75s (0.069hr) 95Mo 96Tc 0.006766546361289079 369792.0
 896 249.75s (0.069hr) 95Mo 92Nb 1.0226326792250566e-14 110452000000000.0
 897
 898 249.75s (0.069hr) 96Mo 97Tc 2.8021754831302685e-10 82049800000000.0
 899 249.75s (0.069hr) 96Mo 96Tc(meta 1) 141.14222460890036 3090.0
 900 249.75s (0.069hr) 96Mo 96Tc 1.044556869653262 369792.0
 901 249.75s (0.069hr) 96Mo 97Tc(meta 1) 0.0032124352502421326 7793280.0
 902 249.75s (0.069hr) 96Mo 93Nb(meta 1) 9.393641087502887e-08 508698000.0
 903
 904 249.75s (0.069hr) 97Mo 97Tc 4.89903113668517e-09 82049800000000.0
 905 249.75s (0.069hr) 97Mo 94Nb(meta 1) 0.0060476562997528556 375.6
 906 249.75s (0.069hr) 97Mo 94Nb 1.7748131840298384e-12 630720000000.0
 907 249.75s (0.069hr) 97Mo 97Tc(meta 1) 0.01793415856007896 7793280.0
 908 249.75s (0.069hr) 97Mo 98Tc 4.155547497613418e-12 135695000000000.0
 909
 910
 911
 912
 913
 914 #####
 915 Time Target Residual Activity Half-life/s
 916 #####
 917
 918 300.0s (0.083hr) 54Fe 55Co 22.289184886562694 63108.0
 919 300.0s (0.083hr) 54Fe 55Fe 2.674825884512542e-05 86686900.0
 920
 921 300.0s (0.083hr) 56Fe 57Co 4.259623845535102 23483500.0
 922 300.0s (0.083hr) 56Fe 53Mn 1.7910361621909062e-11 11613200000000.0
 923
 924 300.0s (0.083hr) 57Fe 58Co(meta 1) 4.6106973008040555 32760.0
 925 300.0s (0.083hr) 57Fe 58Co 0.23384789026830938 6121440.0
 926 300.0s (0.083hr) 57Fe 54Mn 0.001235226887171251 26973200.0
 927 300.0s (0.083hr) 57Fe 57Co 32.662297233133316 23483500.0
 928
 929 300.0s (0.083hr) 58Fe 58Co(meta 1) 359.33092538932937 32760.0
 930 300.0s (0.083hr) 58Fe 58Co 14.803591567872898 6121440.0
 931
 932 300.0s (0.083hr) 12C 13N 0.0022539823826422723 598.02
 933
 934 300.0s (0.083hr) 13C 13N 11761.06146902423 598.02
 935
 936 300.0s (0.083hr) 63Cu 63Zn 466.40230299114216 2299.8

937						
938	300.0s (0.083hr)	65Cu	65Zn	0.11467185018960104	21082500.0	
939						
940	300.0s (0.083hr)	55Mn	55Fe	9.026339726234497	86686900.0	
941						
942						
943	300.0s (0.083hr)	60Ni	61Cu	11.915321731382196	12117.6	
944	300.0s (0.083hr)	60Ni	57Co	3.1671297664530146e-05	23483500.0	
945						
946	300.0s (0.083hr)	61Ni	58Co(meta 1)	0.0024521645481107514	32760.0	
947	300.0s (0.083hr)	61Ni	58Co	2.690674917959629e-05	6121440.0	
948	300.0s (0.083hr)	61Ni	62Cu	15.799446670291239	585.0	
949	300.0s (0.083hr)	61Ni	61Cu	119.90704197644182	12117.6	
950						
951	300.0s (0.083hr)	62Ni	62Cu	551.9287285096955	585.0	
952						
953	300.0s (0.083hr)	64Ni	64Cu	53.751721581959615	45721.4	
954	300.0s (0.083hr)	64Ni	64Zn	7.429964442377365e-13	7.25825e+25	
955	300.0s (0.083hr)	64Ni	61Co	0.002299825477480381	5940.0	
956						
957	300.0s (0.083hr)	28Si	29P	8.050395593678346	4.142	
958						
959	300.0s (0.083hr)	29Si	30P	5.8678718354078985	149.88	
960						
961						
962						
963	300.0s (0.083hr)	52Cr	53Mn	1.7949408315595183e-09	116132000000000.0	
964						
965	300.0s (0.083hr)	53Cr	53Mn	4.4350969142791475e-08	116132000000000.0	
966	300.0s (0.083hr)	53Cr	54Mn	0.00023857416487942494	26973200.0	
967						
968	300.0s (0.083hr)	54Cr	54Mn	0.049249527708328356	26973200.0	
969						
970						
971	300.0s (0.083hr)	51V	51Cr	2.1742806874398735	2393630.0	
972						
973						
974						
975	300.0s (0.083hr)	94Mo	95Tc(meta 1)	0.006651872054830488	5270400.0	
976	300.0s (0.083hr)	94Mo	95Tc	0.44520541994874957	72000.0	
977	300.0s (0.083hr)	94Mo	91Nb	2.76068556452053e-08	21459200000.0	
978	300.0s (0.083hr)	94Mo	91Nb(meta 1)	0.00013705644562074192	5261760.0	
979						
980	300.0s (0.083hr)	95Mo	95Tc(meta 1)	0.06794880691399967	5270400.0	
981	300.0s (0.083hr)	95Mo	95Tc	8.983494037369907	72000.0	
982	300.0s (0.083hr)	95Mo	92Nb(meta 1)	6.283657194688268e-05	876960.0	
983	300.0s (0.083hr)	95Mo	96Tc(meta 1)	0.6843712517853651	3090.0	
984	300.0s (0.083hr)	95Mo	96Tc	0.008158945988498494	369792.0	
985	300.0s (0.083hr)	95Mo	92Nb	1.2283876026727408e-14	1104520000000000.0	
986						
987	300.0s (0.083hr)	96Mo	97Tc	3.3659765563126346e-10	82049800000000.0	
988	300.0s (0.083hr)	96Mo	96Tc(meta 1)	168.59712836012204	3090.0	
989	300.0s (0.083hr)	96Mo	96Tc	1.2623858552993041	369792.0	
990	300.0s (0.083hr)	96Mo	97Tc(meta 1)	0.0038587724583517054	7793280.0	
991	300.0s (0.083hr)	96Mo	93Nb(meta 1)	1.1283652570691754e-07	508698000.0	
992						
993	300.0s (0.083hr)	97Mo	97Tc	5.884722086108313e-09	82049800000000.0	
994	300.0s (0.083hr)	97Mo	94Nb(meta 1)	0.0069624357569702455	375.6	
995	300.0s (0.083hr)	97Mo	94Nb	2.310861793346944e-12	630720000000.0	
996	300.0s (0.083hr)	97Mo	97Tc(meta 1)	0.021542484664906172	7793280.0	
997	300.0s (0.083hr)	97Mo	98Tc	4.9916486457818834e-12	1356950000000000.0	

998
 999
 1000
 1001
 1002
 1003 #####
 1004 Time Target Residual Activity Half-life/s
 1005 #####
 1006
 1007 86001.0s (23.889hr) 54Fe 55Co 8.695485791859939 63108.0
 1008 86001.0s (23.889hr) 54Fe 55Fe 0.009919018911598666 86686900.0
 1009
 1010 86001.0s (23.889hr) 56Fe 57Co 4.248862400311241 23483500.0
 1011 86001.0s (23.889hr) 56Fe 53Mn 1.791036161274762e-11 116132000000000.0
 1012
 1013 86001.0s (23.889hr) 57Fe 58Co(meta 1) 0.7520806650206214 32760.0
 1014 86001.0s (23.889hr) 57Fe 58Co 0.2521111916219682 6121440.0
 1015 86001.0s (23.889hr) 57Fe 54Mn 0.0012325095253074784 26973200.0
 1016 86001.0s (23.889hr) 57Fe 57Co 32.579779730342956 23483500.0
 1017
 1018 86001.0s (23.889hr) 58Fe 58Co(meta 1) 58.61279188337828 32760.0
 1019 86001.0s (23.889hr) 58Fe 58Co 16.259965861785876 6121440.0
 1020
 1021 86001.0s (23.889hr) 12C 13N 1.6329359412455045e-46 598.02
 1022
 1023 86001.0s (23.889hr) 13C 13N 8.520501370314096e-40 598.02
 1024
 1025 86001.0s (23.889hr) 63Cu 63Zn 2.8249771788938045e-09 2299.8
 1026
 1027 86001.0s (23.889hr) 65Cu 65Zn 0.11434919819296589 21082500.0
 1028
 1029 86001.0s (23.889hr) 55Mn 55Fe 9.020156419962703 86686900.0
 1030
 1031
 1032 86001.0s (23.889hr) 60Ni 61Cu 0.08852973508987912 12117.6
 1033 86001.0s (23.889hr) 60Ni 57Co 3.159128380712283e-05 23483500.0
 1034
 1035 86001.0s (23.889hr) 61Ni 58Co(meta 1) 0.0003999884233912975 32760.0
 1036 86001.0s (23.889hr) 61Ni 58Co 3.756117465823082e-05 6121440.0
 1037 86001.0s (23.889hr) 61Ni 62Cu 1.2546407363270588e-43 585.0
 1038 86001.0s (23.889hr) 61Ni 61Cu 0.8908981982103821 12117.6
 1039
 1040 86001.0s (23.889hr) 62Ni 62Cu 4.3828893554833387e-42 585.0
 1041
 1042 86001.0s (23.889hr) 64Ni 64Cu 14.660083227726215 45721.4
 1043 86001.0s (23.889hr) 64Ni 64Zn 7.429964537133324e-13 7.25825e+25
 1044 86001.0s (23.889hr) 64Ni 61Co 1.0435205328684376e-07 5940.0
 1045
 1046 86001.0s (23.889hr) 28Si 29P 0.0 4.142
 1047
 1048 86001.0s (23.889hr) 29Si 30P 4.3681782556436495e-172 149.88
 1049
 1050
 1051
 1052 86001.0s (23.889hr) 52Cr 53Mn 1.7949408306413768e-09 116132000000000.0
 1053
 1054 86001.0s (23.889hr) 53Cr 53Mn 4.435096912010523e-08 116132000000000.0
 1055 86001.0s (23.889hr) 53Cr 54Mn 0.00023804932823276695 26973200.0
 1056

1057	86001.0s (23.889hr)	54Cr	54Mn	0.04914118422115742	26973200.0
1058					
1059					
1060	86001.0s (23.889hr)	51V	51Cr	2.1209849987211724	2393630.0
1061					
1062					
1063					
1064	86001.0s (23.889hr)	94Mo	95Tc(meta 1)	0.0065773188250607085	5270400.0
1065	86001.0s (23.889hr)	94Mo	95Tc	0.19523943137423416	72000.0
1066	86001.0s (23.889hr)	94Mo	91Nb	2.7974990678177886e-08	21459200000.0
1067	86001.0s (23.889hr)	94Mo	91Nb(meta 1)	0.00013551782846282147	5261760.0
1068					
1069	86001.0s (23.889hr)	95Mo	95Tc(meta 1)	0.06718724641303317	5270400.0
1070	86001.0s (23.889hr)	95Mo	95Tc	3.9381668369886524	72000.0
1071	86001.0s (23.889hr)	95Mo	92Nb(meta 1)	5.872111574303616e-05	876960.0
1072	86001.0s (23.889hr)	95Mo	96Tc(meta 1)	3.0636520637167826e-09	3090.0
1073	86001.0s (23.889hr)	95Mo	96Tc	0.011760947971286923	369792.0
1074	86001.0s (23.889hr)	95Mo	92Nb	1.2283876026066756e-14	1104520000000000.0
1075					
1076	86001.0s (23.889hr)	96Mo	97Tc	3.393807557284754e-10	82049800000000.0
1077	86001.0s (23.889hr)	96Mo	96Tc(meta 1)	7.547408499257139e-07	3090.0
1078	86001.0s (23.889hr)	96Mo	96Tc	2.260695558186092	369792.0
1079	86001.0s (23.889hr)	96Mo	97Tc(meta 1)	0.0038294712108109985	7793280.0
1080	86001.0s (23.889hr)	96Mo	93Nb(meta 1)	1.128233499471212e-07	508698000.0
1081					
1082	86001.0s (23.889hr)	97Mo	97Tc	5.900259379540796e-09	82049800000000.0
1083	86001.0s (23.889hr)	97Mo	94Nb(meta 1)	1.4337326674050271e-71	375.6
1084	86001.0s (23.889hr)	97Mo	94Nb	6.436329568549951e-12	630720000000.0
1085	86001.0s (23.889hr)	97Mo	97Tc(meta 1)	0.021378903711994034	7793280.0
1086	86001.0s (23.889hr)	97Mo	98Tc	4.991648643596682e-12	135695000000000.0
1087					
1088					
1089					
1090					
1091					
1092	#####	#####	#####	#####	#####
1093	Time	Target	Residual	Activity	Half-life/s
1094	#####	#####	#####	#####	#####
1095					
1096	173000.5s (48.056hr)	54Fe	55Co	3.3442564122033365	63108.0
1097	173000.5s (48.056hr)	54Fe	55Fe	0.013806245058191325	86686900.0
1098					
1099	173000.5s (48.056hr)	56Fe	57Co	4.2379657110936195	23483500.0
1100	173000.5s (48.056hr)	56Fe	53Mn	1.7910361603447373e-11	116132000000000.0
1101					
1102	173000.5s (48.056hr)	57Fe	58Co(meta 1)	0.11935218799048451	32760.0
1103	173000.5s (48.056hr)	57Fe	58Co	0.2530045030032729	6121440.0
1104	173000.5s (48.056hr)	57Fe	54Mn	0.0012297571056677618	26973200.0
1105	173000.5s (48.056hr)	57Fe	57Co	32.49622519243321	23483500.0
1106					
1107	173000.5s (48.056hr)	58Fe	58Co(meta 1)	9.301615213469551	32760.0
1108	173000.5s (48.056hr)	58Fe	58Co	16.36279843820864	6121440.0
1109					
1110	173000.5s (48.056hr)	12C	13N	2.626344531585545e-90	598.02
1111					
1112	173000.5s (48.056hr)	13C	13N	1.3704011048481893e-83	598.02
1113					
1114	173000.5s (48.056hr)	63Cu	63Zn	1.1569200743006014e-20	2299.8
1115					

1116	173000.5s (48.056hr)	65Cu	65Zn	0.11402258609895961	21082500.0
1117					
1118	173000.5s (48.056hr)	55Mn	55Fe	9.013883759715732	86686900.0
1119					
1120					
1121	173000.5s (48.056hr)	60Ni	61Cu	0.0006106814505442664	12117.6
1122	173000.5s (48.056hr)	60Ni	57Co	3.1510264379050344e-05	23483500.0
1123					
1124	173000.5s (48.056hr)	61Ni	58Co(meta 1)	6.347656005929455e-05	32760.0
1125	173000.5s (48.056hr)	61Ni	58Co	3.898246631850751e-05	6121440.0
1126	173000.5s (48.056hr)	61Ni	62Cu	2.139011981303592e-88	585.0
1127	173000.5s (48.056hr)	61Ni	61Cu	0.0061454493613704135	12117.6
1128					
1129	173000.5s (48.056hr)	62Ni	62Cu	7.472300693465575e-87	585.0
1130					
1131	173000.5s (48.056hr)	64Ni	64Cu	3.920407042818884	45721.4
1132	173000.5s (48.056hr)	64Ni	64Zn	7.429964563165702e-13	7.25825e+25
1133	173000.5s (48.056hr)	64Ni	61Co	4.069126104613427e-12	5940.0
1134					
1135	173000.5s (48.056hr)	28Si	29P	0.0	4.142
1136					
1137	173000.5s (48.056hr)	29Si	30P	0.0	149.88
1138					
1139					
1140					
1141	173000.5s (48.056hr)	52Cr	53Mn	1.7949408297093244e-09	116132000000000.0
1142					
1143	173000.5s (48.056hr)	53Cr	53Mn	4.435096909707526e-08	116132000000000.0
1144	173000.5s (48.056hr)	53Cr	54Mn	0.0002375177204579015	26973200.0
1145					
1146	173000.5s (48.056hr)	54Cr	54Mn	0.04903144295117779	26973200.0
1147					
1148					
1149	173000.5s (48.056hr)	51V	51Cr	2.0682178505494626	2393630.0
1150					
1151					
1152					
1153	173000.5s (48.056hr)	94Mo	95Tc(meta 1)	0.006502490620724443	5270400.0
1154	173000.5s (48.056hr)	94Mo	95Tc	0.08463765809831882	72000.0
1155	173000.5s (48.056hr)	94Mo	91Nb	2.8344474693315093e-08	21459200000.0
1156	173000.5s (48.056hr)	94Mo	91Nb(meta 1)	0.00013397356522212487	5261760.0
1157					
1158	173000.5s (48.056hr)	95Mo	95Tc(meta 1)	0.06642287704960351	5270400.0
1159	173000.5s (48.056hr)	95Mo	95Tc	1.7057909975131167	72000.0
1160	173000.5s (48.056hr)	95Mo	92Nb(meta 1)	5.4818908450236494e-05	876960.0
1161	173000.5s (48.056hr)	95Mo	96Tc(meta 1)	1.0249114194195995e-17	3090.0
1162	173000.5s (48.056hr)	95Mo	96Tc	0.009991258471636892	369792.0
1163	173000.5s (48.056hr)	95Mo	92Nb	1.2283876025396093e-14	1104520000000000.0
1164					
1165	173000.5s (48.056hr)	96Mo	97Tc	3.4218440888468736e-10	82049800000000.0
1166	173000.5s (48.056hr)	96Mo	96Tc(meta 1)	2.5249032843921147e-15	3090.0
1167	173000.5s (48.056hr)	96Mo	96Tc	1.9205249199593584	369792.0
1168	173000.5s (48.056hr)	96Mo	97Tc(meta 1)	0.003799953574939192	7793280.0
1169	173000.5s (48.056hr)	96Mo	93Nb(meta 1)	1.1280997612825722e-07	508698000.0
1170					
1171	173000.5s (48.056hr)	97Mo	97Tc	5.915911415107868e-09	82049800000000.0
1172	173000.5s (48.056hr)	97Mo	94Nb(meta 1)	2.688275357702702e-141	375.6
1173	173000.5s (48.056hr)	97Mo	94Nb	6.436328953169255e-12	630720000000.0
1174	173000.5s (48.056hr)	97Mo	97Tc(meta 1)	0.021214114721459905	7793280.0
1175	173000.5s (48.056hr)	97Mo	98Tc	4.991648641378372e-12	135695000000000.0
1176					

1177
 1178
 1179
 1180
 1181 #####

1182	Time	Target	Residual	Activity	Half-life/s
1183	#####	#####	#####	#####	#####
1184					
1185	260000.0s (72.222hr)	54Fe	55Co	1.286190469201019	63108.0
1186	260000.0s (72.222hr)	54Fe	55Fe	0.015294311892069673	86686900.0
1187					
1188	260000.0s (72.222hr)	56Fe	57Co	4.227096967670594	23483500.0
1189	260000.0s (72.222hr)	56Fe	53Mn	1.791036159414712e-11	116132000000000.0
1190					
1191	260000.0s (72.222hr)	57Fe	58Co(meta 1)	0.018940713996051697	32760.0
1192	260000.0s (72.222hr)	57Fe	58Co	0.25105831080080615	6121440.0
1193	260000.0s (72.222hr)	57Fe	54Mn	0.0012270108326855093	26973200.0
1194	260000.0s (72.222hr)	57Fe	57Co	32.41288493960651	23483500.0
1195					
1196	260000.0s (72.222hr)	58Fe	58Co(meta 1)	1.4761290632870192	32760.0
1197	260000.0s (72.222hr)	58Fe	58Co	16.244011041791023	6121440.0
1198					
1199	260000.0s (72.222hr)	12C	13N	4.2241005445249467e-134	598.02
1200					
1201	260000.0s (72.222hr)	13C	13N	2.20409469648376e-127	598.02
1202					
1203	260000.0s (72.222hr)	63Cu	63Zn	4.7379641447008975e-32	2299.8
1204					
1205	260000.0s (72.222hr)	65Cu	65Zn	0.11369690689702112	21082500.0
1206					
1207	260000.0s (72.222hr)	55Mn	55Fe	9.00761546150693	86686900.0
1208					
1209					
1210	260000.0s (72.222hr)	60Ni	61Cu	4.212503670774942e-06	12117.6
1211	260000.0s (72.222hr)	60Ni	57Co	3.1429452734484396e-05	23483500.0
1212					
1213	260000.0s (72.222hr)	61Ni	58Co(meta 1)	1.0073475734120185e-05	32760.0
1214	260000.0s (72.222hr)	61Ni	58Co	3.888431409471208e-05	6121440.0
1215	260000.0s (72.222hr)	61Ni	62Cu	3.646758887771052e-133	585.0
1216	260000.0s (72.222hr)	61Ni	61Cu	4.239154140061418e-05	12117.6
1217					
1218	260000.0s (72.222hr)	62Ni	62Cu	1.2739376499137902e-131	585.0
1219					
1220	260000.0s (72.222hr)	64Ni	64Cu	1.0483972800588075	45721.4
1221	260000.0s (72.222hr)	64Ni	64Zn	7.429964570127294e-13	7.25825e+25
1222	260000.0s (72.222hr)	64Ni	61Co	1.5867236660626395e-16	5940.0
1223					
1224	260000.0s (72.222hr)	28Si	29P	0.0	4.142
1225					
1226	260000.0s (72.222hr)	29Si	30P	0.0	149.88
1227					
1228					
1229					
1230	260000.0s (72.222hr)	52Cr	53Mn	1.7949408287772715e-09	116132000000000.0
1231					
1232	260000.0s (72.222hr)	53Cr	53Mn	4.435096907404529e-08	116132000000000.0
1233	260000.0s (72.222hr)	53Cr	54Mn	0.00023698729986061976	26973200.0
1234					
1235	260000.0s (72.222hr)	54Cr	54Mn	0.048921946753565204	26973200.0

1236
 1237
 1238 260000.0s (72.222hr) 51V 51Cr 2.016763475418512 2393630.0
 1239
 1240
 1241
 1242 260000.0s (72.222hr) 94Mo 95Tc(meta 1) 0.006428513714662307 5270400.0
 1243 260000.0s (72.222hr) 94Mo 95Tc 0.036770852278538234 72000.0
 1244 260000.0s (72.222hr) 94Mo 91Nb 2.8709746401498208e-08 21459200000.0
 1245 260000.0s (72.222hr) 94Mo 91Nb(meta 1) 0.00013244689928935163 5261760.0
 1246
 1247 260000.0s (72.222hr) 95Mo 95Tc(meta 1) 0.06566720368958137 5270400.0
 1248 260000.0s (72.222hr) 95Mo 95Tc 0.7396681801118211 72000.0
 1249 260000.0s (72.222hr) 95Mo 92Nb(meta 1) 5.117601540178146e-05 876960.0
 1250 260000.0s (72.222hr) 95Mo 96Tc(meta 1) 3.4287294895436346e-26 3090.0
 1251 260000.0s (72.222hr) 95Mo 96Tc 0.00848785709077453 369792.0
 1252 260000.0s (72.222hr) 95Mo 92Nb 1.2283876024725429e-14 1104520000000000.0
 1253
 1254 260000.0s (72.222hr) 96Mo 97Tc 3.4496645142583677e-10 82049800000000.0
 1255 260000.0s (72.222hr) 96Mo 96Tc(meta 1) 8.446788849658192e-24 3090.0
 1256 260000.0s (72.222hr) 96Mo 96Tc 1.6315403215882949 369792.0
 1257 260000.0s (72.222hr) 96Mo 97Tc(meta 1) 0.0037706634615579594 7793280.0
 1258 260000.0s (72.222hr) 96Mo 93Nb(meta 1) 1.1279660389469478e-07 508698000.0
 1259
 1260 260000.0s (72.222hr) 97Mo 97Tc 5.931442804446482e-09 82049800000000.0
 1261 260000.0s (72.222hr) 97Mo 94Nb(meta 1) 5.040566183032934e-211 375.6
 1262 260000.0s (72.222hr) 97Mo 94Nb 6.436328337788618e-12 630720000000.0
 1263 260000.0s (72.222hr) 97Mo 97Tc(meta 1) 0.021050595927553588 7793280.0
 1264 260000.0s (72.222hr) 97Mo 98Tc 4.99164863916006e-12 1356950000000000.0
 1265
 1266
 1267
 1268
 1269
 1270 2108.681 Gamma Dose - Beam End
 1271 2108.681 ======
 1272 2108.681 Activity (Bq) 1.3624e+04
 1273 2108.681 Gamma Energy (eV) 1.5362e+08
 1274 2108.681 Gamma Energy (mW) 2.4613e-08
 1275 2108.681 Absorbed Dose (mGy/s) 2.4475e-17
 1276 2108.681 Absorbed Dose (mGy/hr) 8.8112e-14
 1277 2108.681
 1278 2108.681
 1279 2108.681 Gamma Dose - Sim End
 1280 2108.681 ======
 1281 2108.681 Activity (Bq) 7.0682e+01
 1282 2108.681 Gamma Energy (eV) 2.4486e+07
 1283 2108.681 Gamma Energy (mW) 3.9231e-09
 1284 2108.681 Absorbed Dose (mGy/s) 3.9013e-18
 1285 2108.681 Absorbed Dose (mGy/hr) 1.4045e-14
 1286 2108.681
 1287 2108.681
 1288 2108.681 Absorbed Dose Calculations
 1289 2108.681 ======
 1290 2108.681 Absorbed dose assumptions:
 1291 2108.681 1. radiation from point, emitted isotropically
 1292 2108.681 2. 80Kg human
 1293 2108.681 3. 1m from point source
 1294 2108.681 4. 1m squared surface area
 1295 2108.681 5. all energy absorbed
 1296 2108.681

1297 2108.681
1298 2108.681 Dose Limits
1299 2108.681 =====
1300 2108.681 employees 18+ 20 millisieverts/year
1301 2108.681 trainees 18+ 6 millisieverts/year
1302 2108.681 public and under 18s 1 millisievert/year
1303 2108.681 public and under 18s 1.140771128E-04 millisieverts/hour
1304 2108.681 public and under 18s 3.2E-08 millisieverts/second
1305 2108.681
1306 2108.681 Dose averaged over area of skin not exceeding 1cm²
1307 2108.681 Source: <http://www.hse.gov.uk/radiation/ionising/doses/>
1308 2108.681
1309 2108.681 1mW gammas completely absorbed by 80kg human will give public limit for entire year over 80 seconds.
1310 2108.681
1311 2108.681 Dose under 1W gammas absorbed for 1s no immediate changes (under 100 Rads)
1312 2108.681 Dose 1W-2W gammas absorbed for 1s acute radiation syndrome (100-200 Rads)
1313 2108.681 Dose 10W+ gammas absorbed for 1s mostly fatal (1000+ Rads)

Appendix F

Iron Activity: Irradiated to 100DPA

F.1 Activity vs Beam Energy

The beam energy was set to 5MeV, 10MeV, 15MeV, 20MeV, 25MeV and 30MeV within SRIM, and the Activity V2 code was used to calculate how radioactive each sample would be if irradiated to 100 DPA of damage.

The beam settings are:

- target material = pure iron
- target thickness = 0.5mm
- beam current = 60 micro amps (maximum proton current for the Scanditronix MC-40)
- beam area = 64mm^2

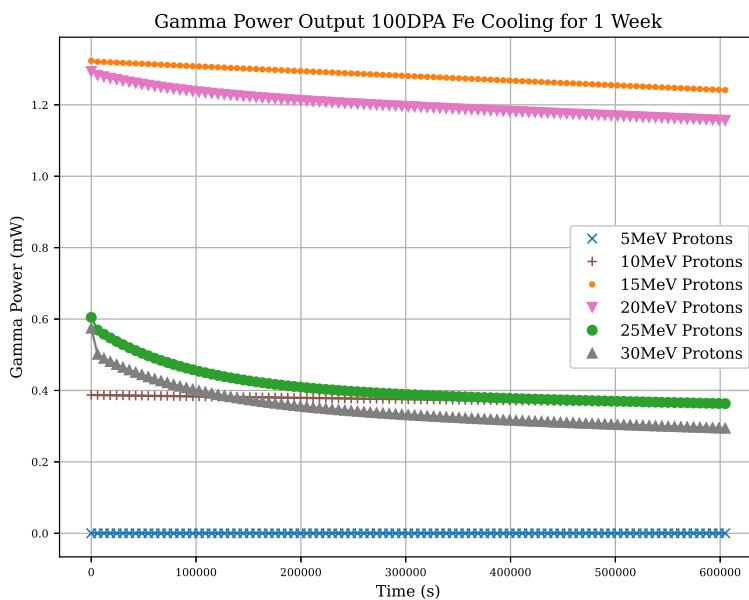


Figure F.1: Cooling measured by gamma power for iron samples irradiated to 100DPA at a range of proton energies

F.1.1 5MeV Proton Beam - 60 microamp 100DPA

With a proton energy of 5MeV the target must be irradiated for 205 days to reach 100DPA.

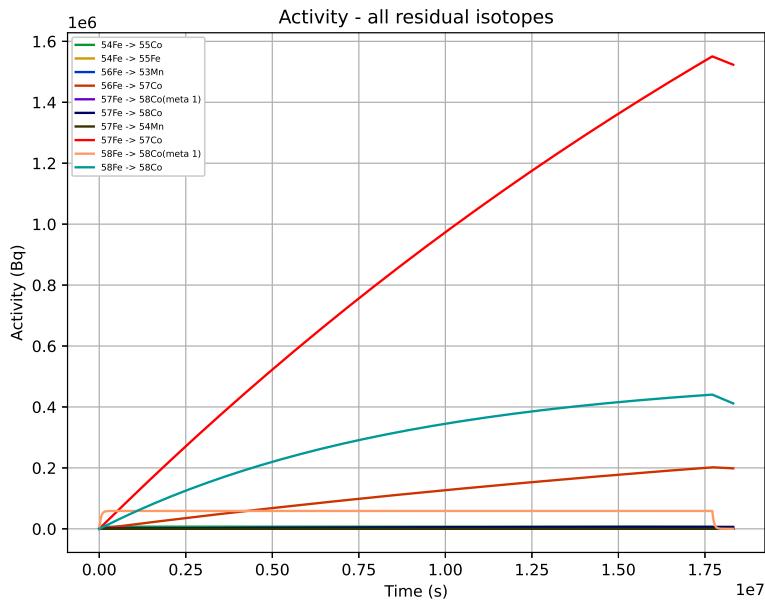


Figure F.2: Residual isotope activity over time with 5MeV protons

Residual gamma power at time t=17712000.0
Total gamma power: 9.435311052144465e-05 mW

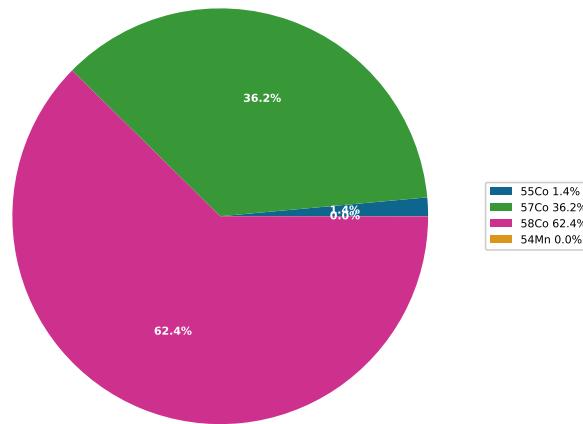


Figure F.3: Residual isotope gamma output at the end of irradiation with 5MeV protons

Residual gamma power at time t=18316800.0
Total gamma power: 8.852972608256112e-05 mW

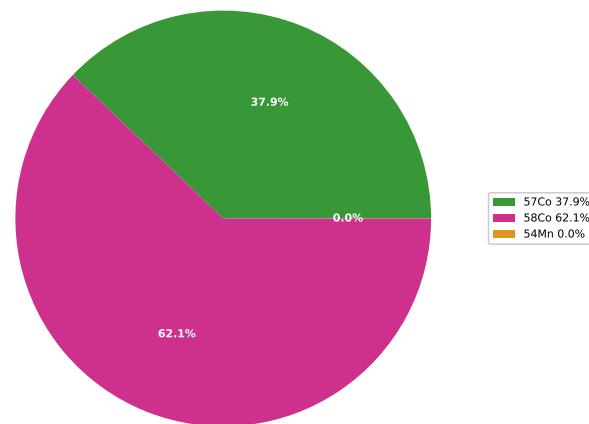


Figure F.4: Residual isotope gamma output after 1 week of cooling

F.1.2 10MeV Proton Beam - 60 microamp 100DPA

The vacancies per ion are at a maximum out of the six settings here with a proton energy of 10MeV. As a result, the target must be irradiated for just 134 days to reach 100DPA.

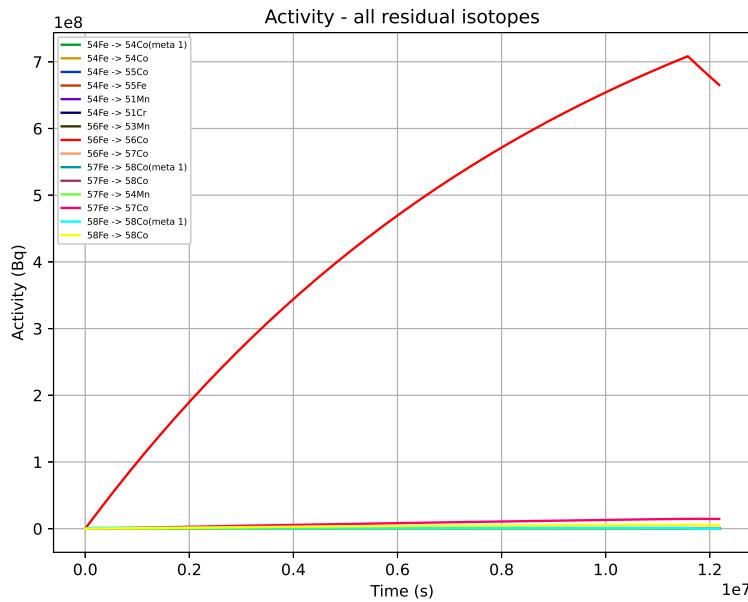


Figure F.5: Residual isotope activity over time with 10MeV protons

Residual gamma power at time t=11577600.0
Total gamma power: 0.3872607000597164 mW

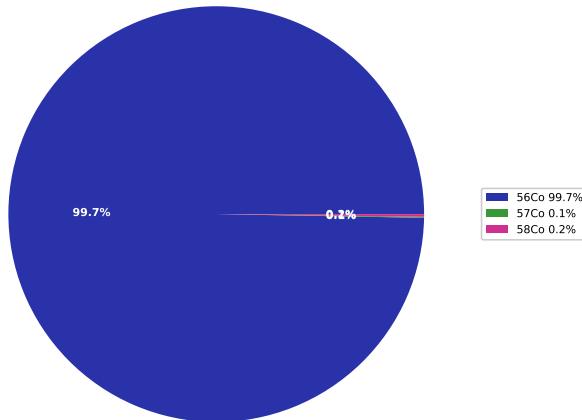


Figure F.6: Residual isotope gamma output at the end of irradiation with 10MeV protons

Residual gamma power at time t=12182400.0
Total gamma power: 0.3636776414065111 mW

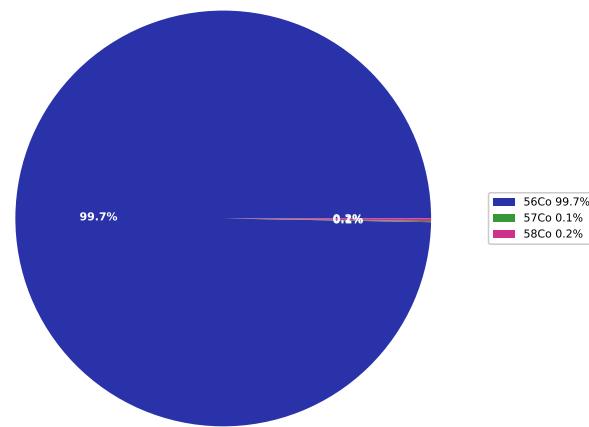


Figure F.7: Residual isotope gamma output after 1 week of cooling

F.1.3 15MeV Proton Beam - 60 microamp 100DPA

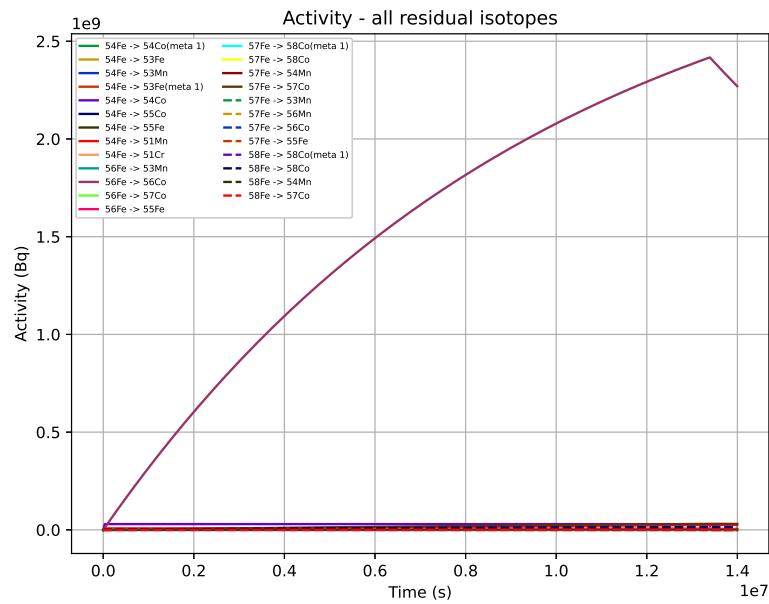


Figure F.8: Residual isotope activity over time with 15MeV protons

Residual gamma power at time t=13392000.0
Total gamma power: 1.3232690955956106 mW

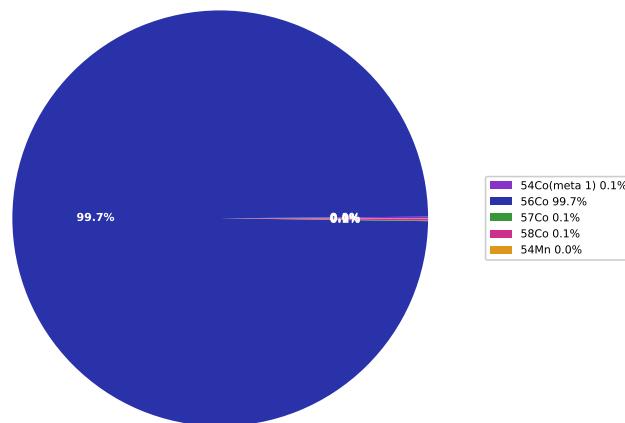


Figure F.9: Residual isotope gamma output at the end of irradiation with 15MeV protons

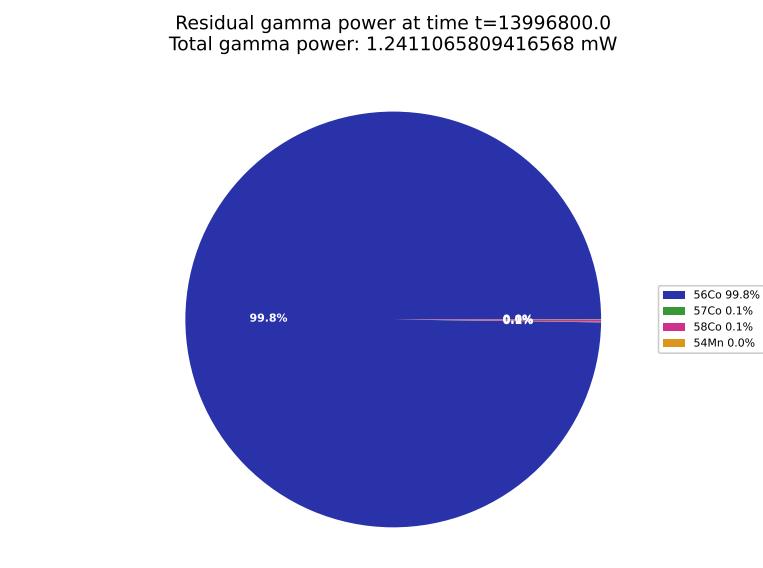


Figure F.10: Residual isotope gamma output after 1 week of cooling

F.1.4 20MeV Proton Beam - 60 microamp 100DPA

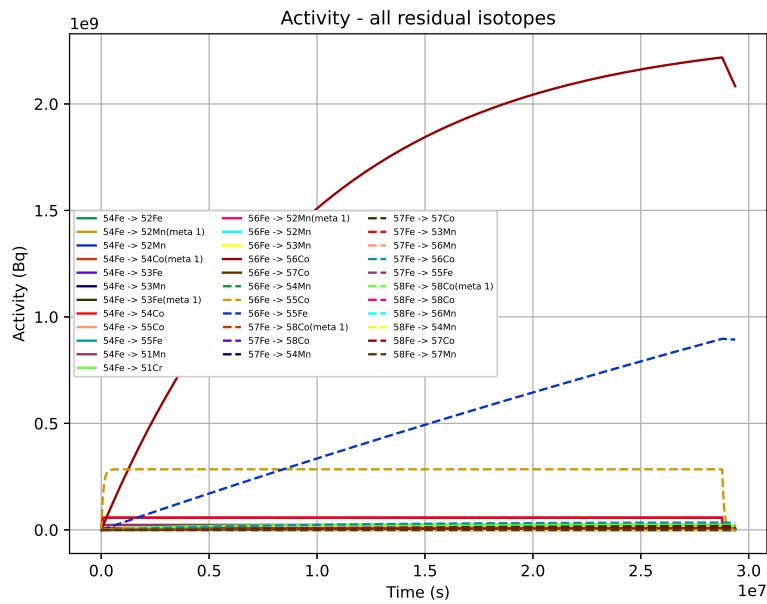


Figure F.11: Residual isotope activity over time with 20MeV protons

Residual gamma power at time t=28771200.0
Total gamma power: 1.2934181154506263 mW

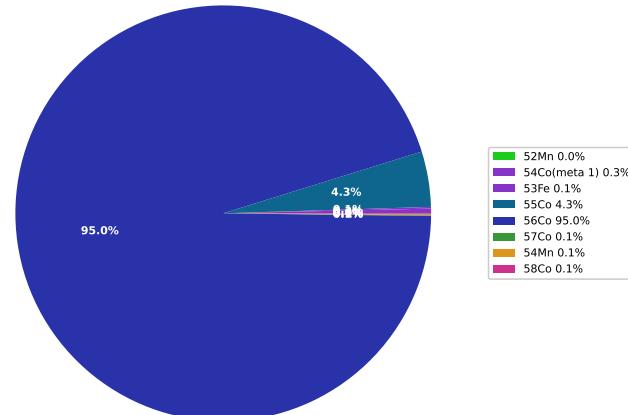


Figure F.12: Residual isotope gamma output at the end of irradiation with 20MeV protons

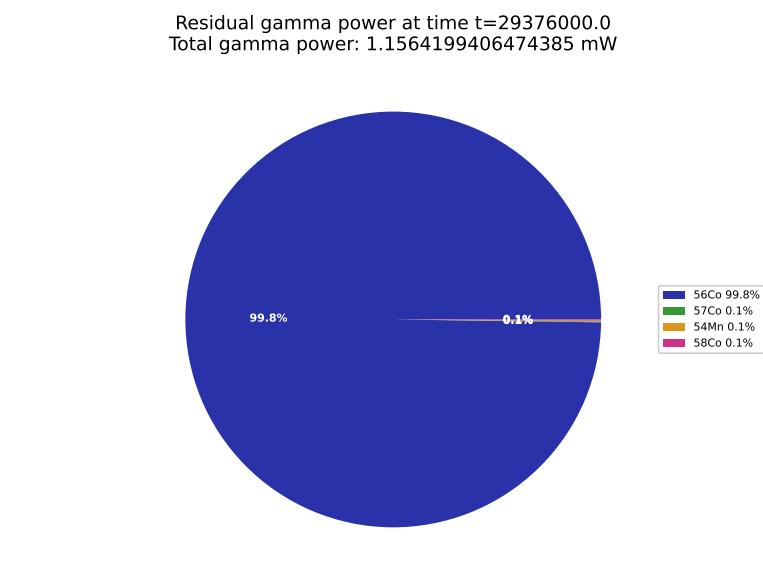


Figure F.13: Residual isotope gamma output after 1 week of cooling

F.1.5 25MeV Proton Beam - 60 microamp 100DPA

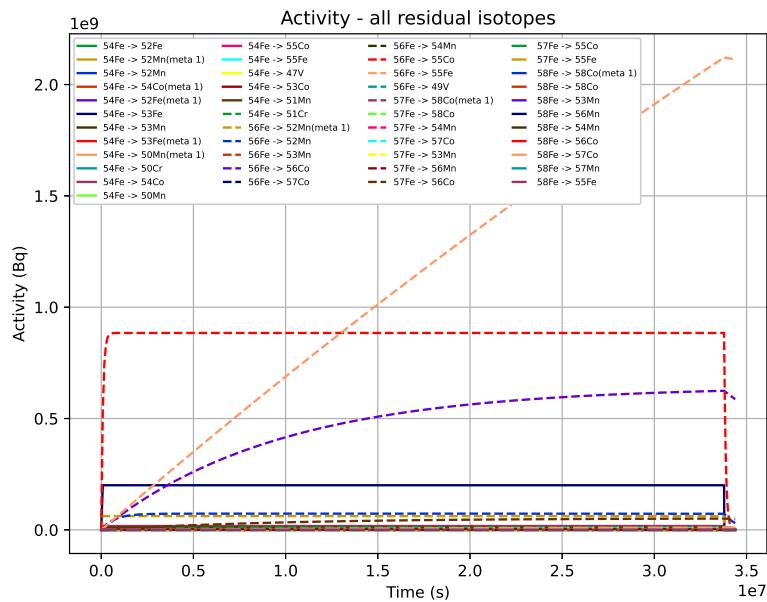


Figure F.14: Residual isotope activity over time with 25MeV protons

Residual gamma power at time $t=33782400.0$
Total gamma power: 0.6047827408353267 mW

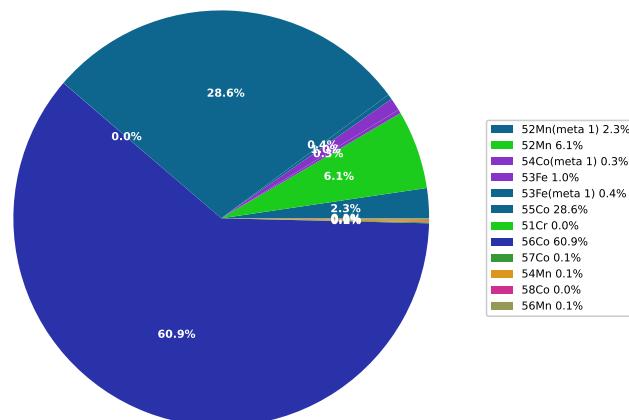


Figure F.15: Residual isotope gamma output at the end of irradiation with 25MeV protons

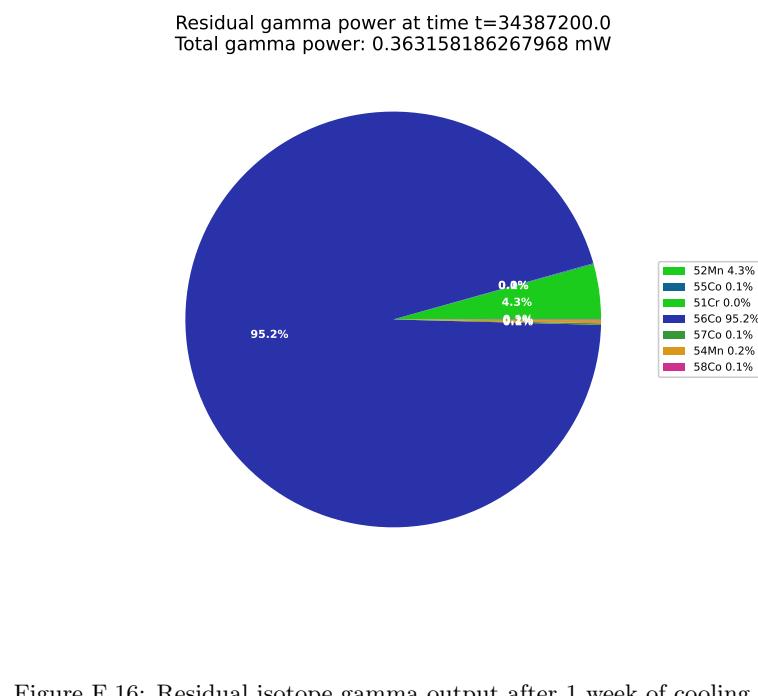


Figure F.16: Residual isotope gamma output after 1 week of cooling

F.1.6 30MeV Proton Beam - 60 microamp 100DPA

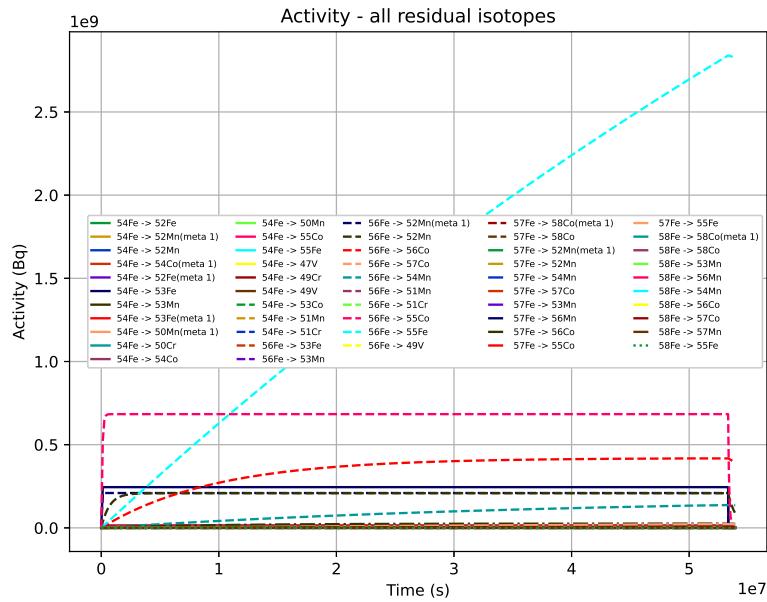


Figure F.17: Residual isotope activity over time with 30MeV protons

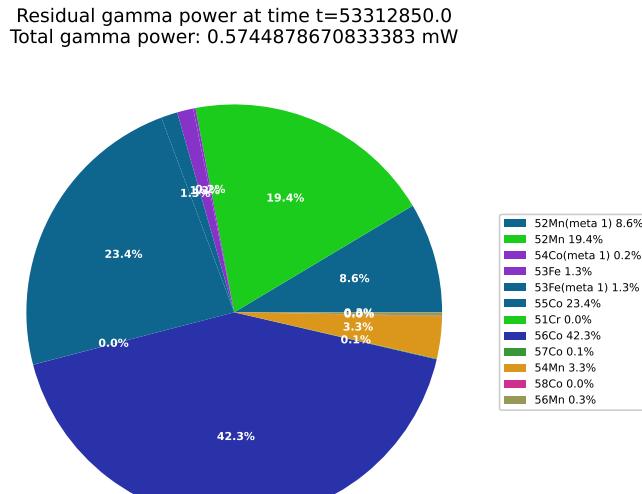


Figure F.18: Residual isotope gamma output at the end of irradiation with 30MeV protons

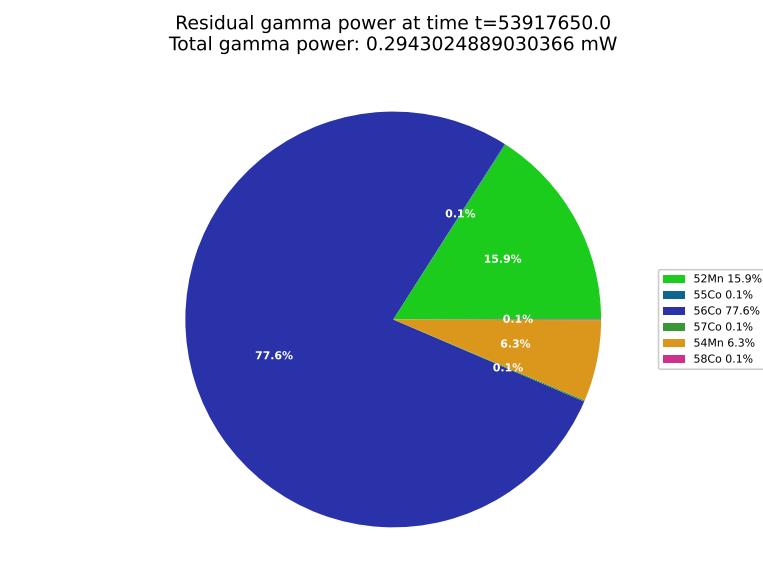


Figure F.19: Residual isotope gamma output after 1 week of cooling

Appendix G

SRIM Results

G.0.1 Ion paths through an iron sample

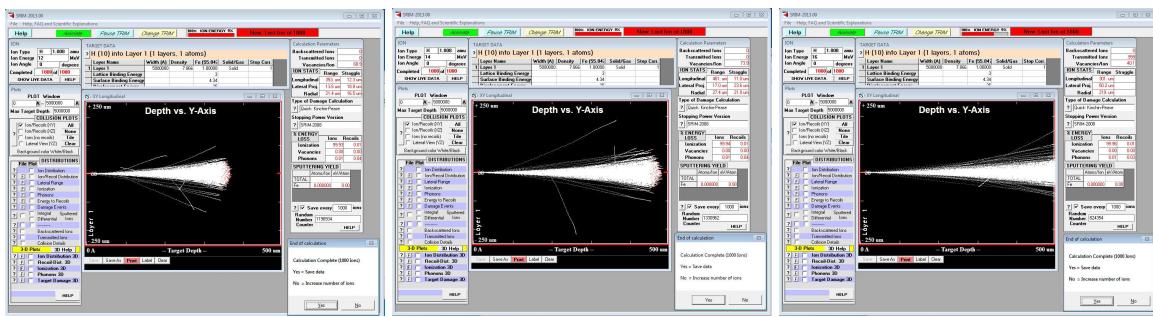


Figure G.1: SRIM trajectories for 12MeV, 14MeV and 16MeV protons in iron

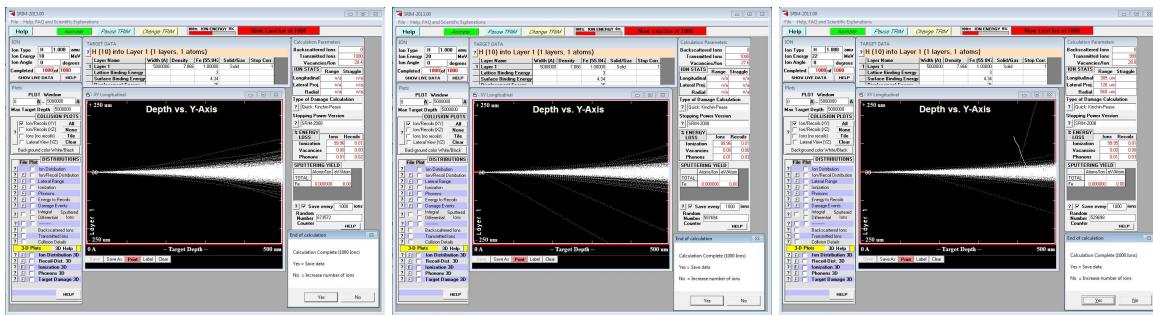


Figure G.2: SRIM trajectories for 18MeV, 20MeV and 22MeV protons in iron

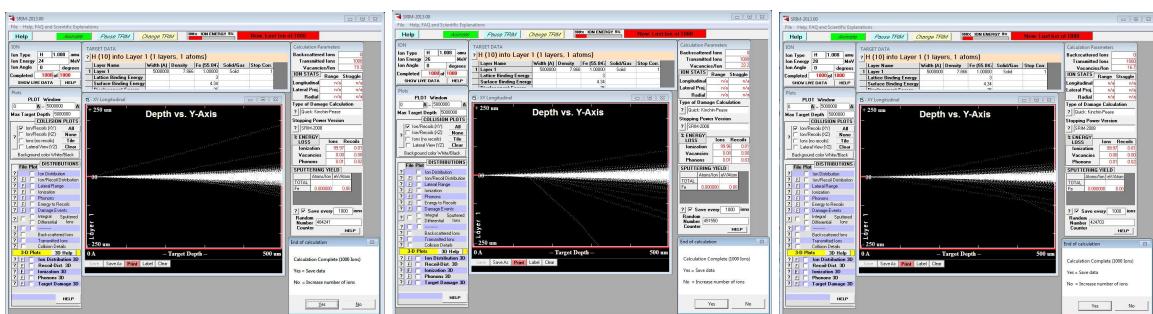


Figure G.3: SRIM trajectories for 24MeV, 26MeV and 28MeV protons in iron

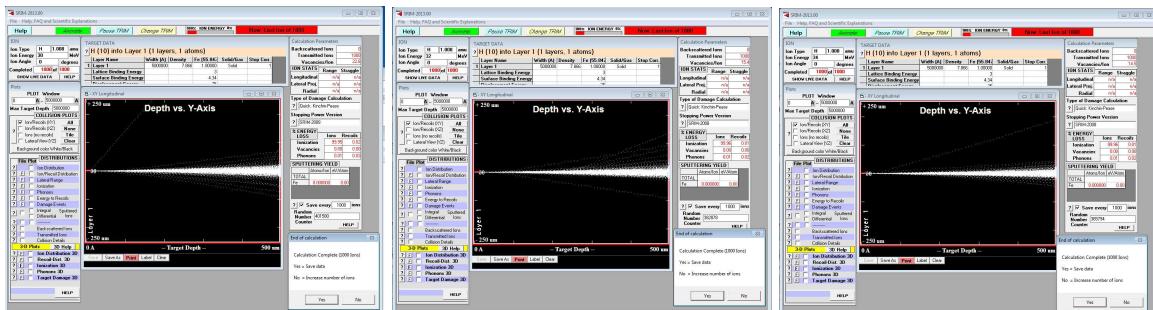


Figure G.4: SRIM trajectories for 30MeV, 32MeV and 34MeV protons in iron

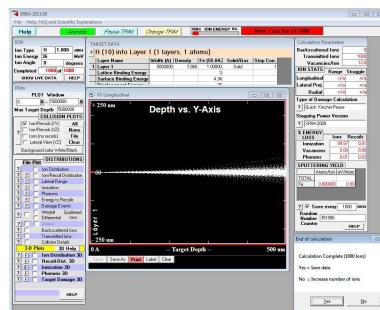


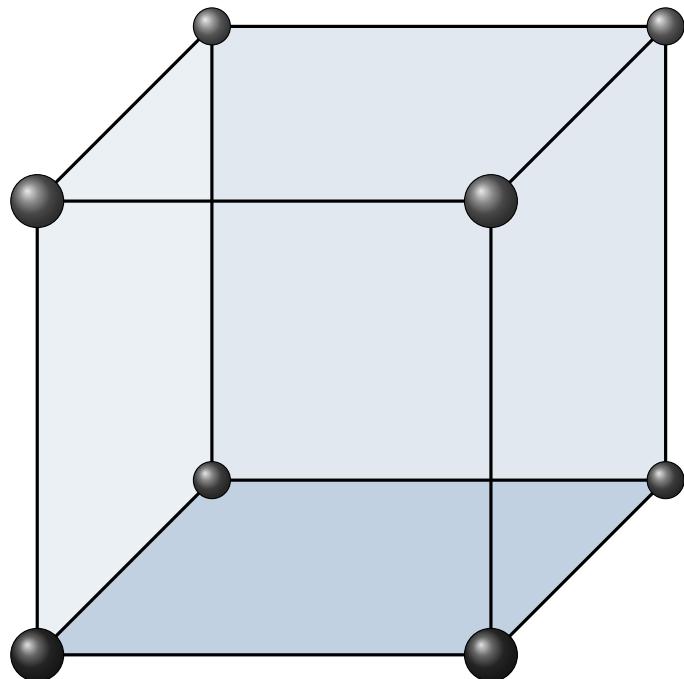
Figure G.5: SRIM trajectories for 36MeV protons in iron

Appendix H

Crystal Structures

H.1 Common Cubic Structures

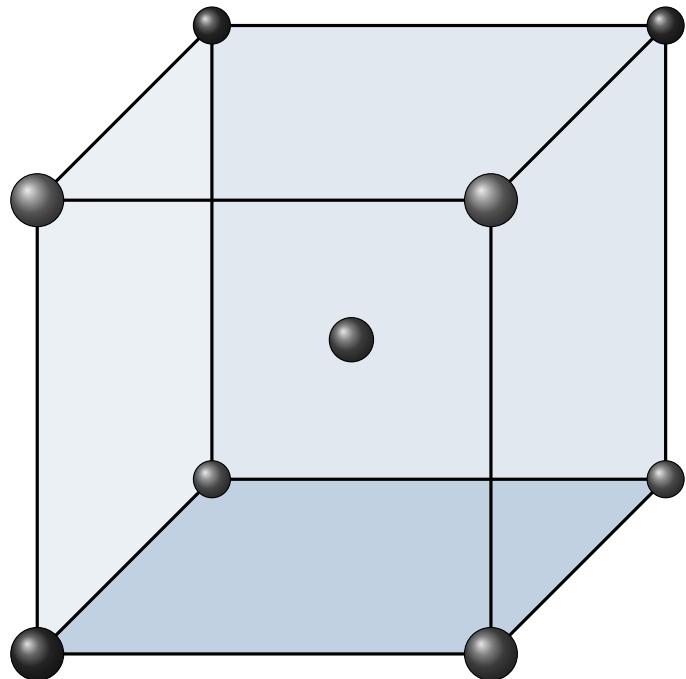
H.1.1 Simple Cubic



Sample Configuration
[0.0 0.0 0.0]

Table H.1: Simple cubic configuration

H.1.2 Body Centered Cubic

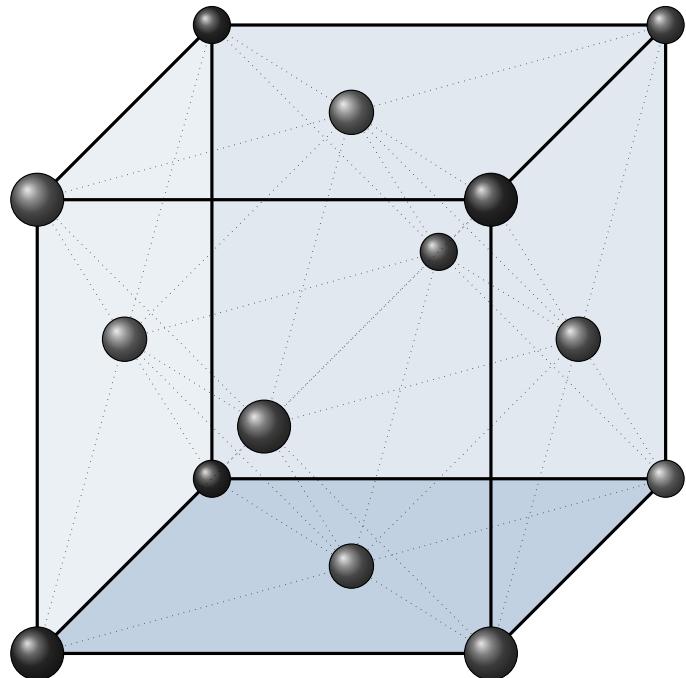


Sample Configuration

$$\begin{bmatrix} 0.0 & 0.0 & 0.0 \\ 0.5 & 0.5 & 0.5 \end{bmatrix}$$

Table H.2: Body centered cubic configuration

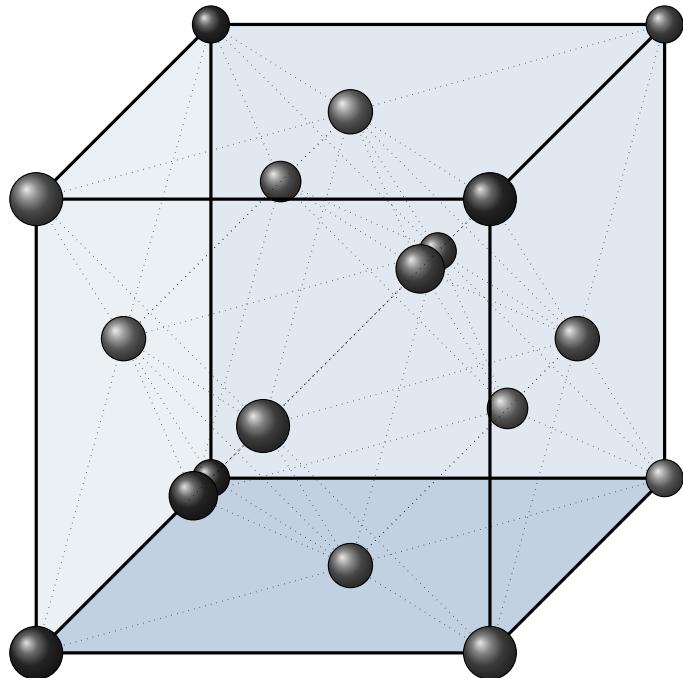
H.1.3 Face Centered Cubic



Sample Configuration

[0.0	0.0	0.0]
[0.0	0.5	0.5]
[0.5	0.0	0.5]
[0.5	0.5	0.0]

Table H.3: Face centered cubic configuration

H.1.4 Zincblende

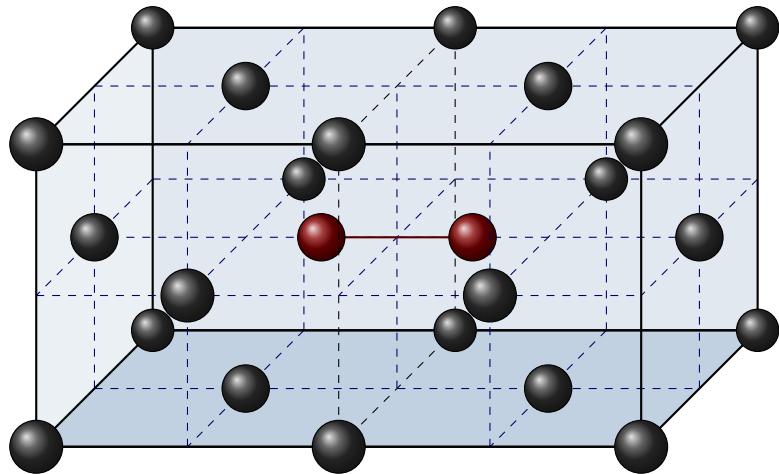
Sample Configuration

[0.0	0.0	0.0]
[0.0	0.5	0.5]
[0.5	0.0	0.5]
[0.5	0.5	0.0]
[0.25	0.25	0.25]
[0.75	0.75	0.25]
[0.25	0.75	0.75]
[0.75	0.25	0.75]

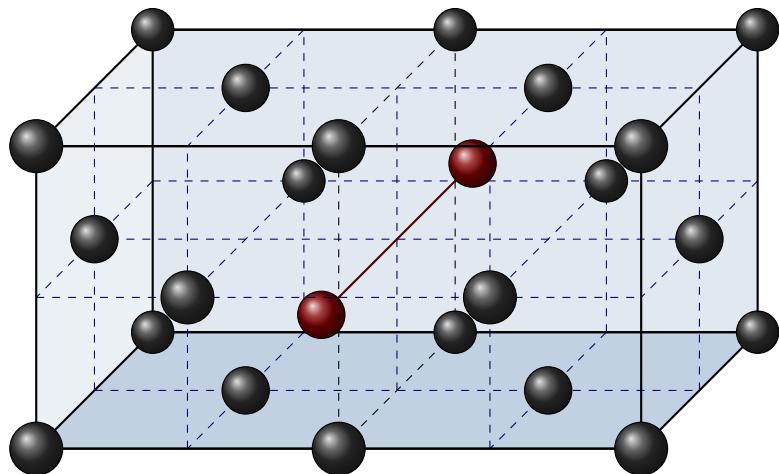
Table H.4: Zincblende configuration

H.2 FCC Defects

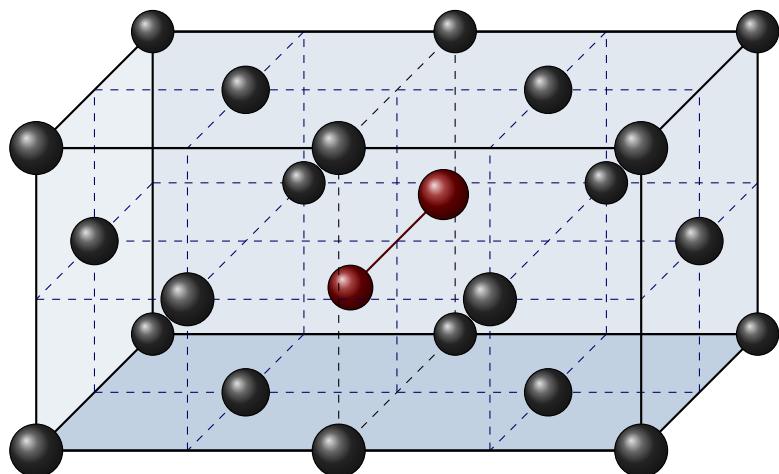
H.2.1 Dumbbell $<100>$



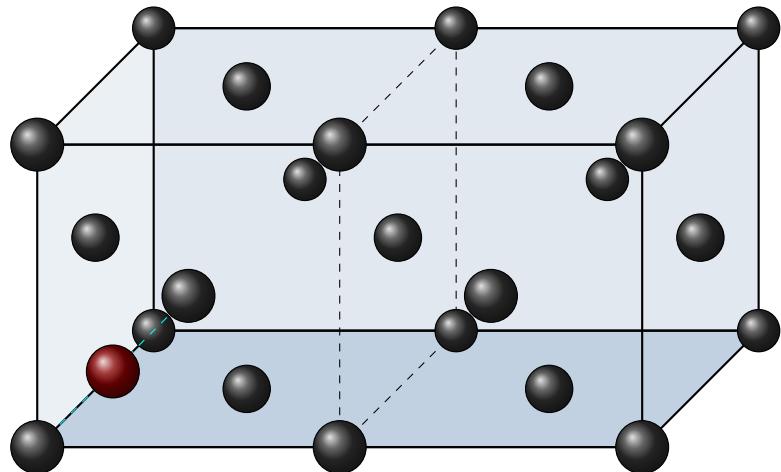
H.2.2 Dumbbell $<110>$



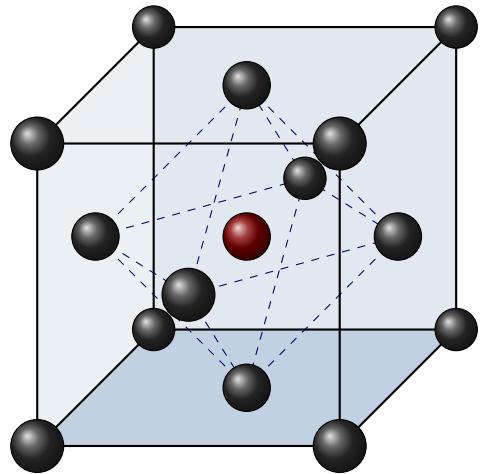
H.2.3 Dumbbell $<111>$



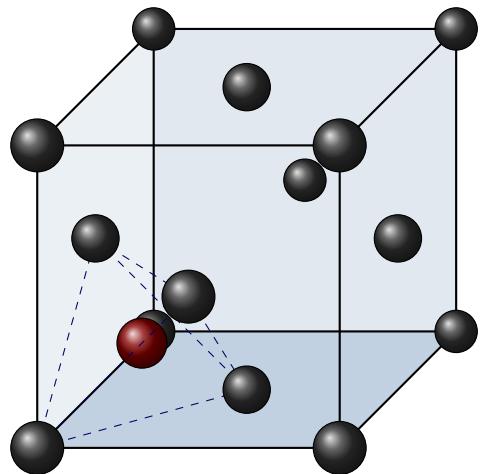
H.2.4 Crowdion



H.2.5 Octahedral Interstitial



H.2.6 Tetrahedral Interstitial



Appendix I

Elastic Constants

I.1 Elastic Constant Matrices

Crystal:	Cubic
Sides:	$a = b = c$
Angles:	$\alpha = \beta = \gamma = \frac{\pi}{2}$ radians
Symmetry:	$\frac{\pi}{2}$ radians rotation, 3 mutually orthogonal planes of symmetry
No. Independent Elastic Constants:	3
Elastic Constants:	$\begin{bmatrix} C_{11} & C_{12} & C_{12} & 0 & 0 & 0 \\ C_{12} & C_{11} & C_{12} & 0 & 0 & 0 \\ C_{12} & C_{12} & C_{11} & 0 & 0 & 0 \\ 0 & 0 & 0 & C_{44} & 0 & 0 \\ 0 & 0 & 0 & 0 & C_{44} & 0 \\ 0 & 0 & 0 & 0 & 0 & C_{44} \end{bmatrix}$

Table I.1: Independent elastic constants for a cubic crystal[223]

Crystal:	Orthorhombic/Orthotropic
Sides:	$a \neq b \neq c$
Angles:	$\alpha = \beta = \gamma = \frac{\pi}{2}$ radians
Symmetry:	3 mutually orthogonal planes of symmetry
No. Independent Elastic Constants:	9
Elastic Constants:	$\begin{bmatrix} C_{11} & C_{12} & C_{13} & 0 & 0 & 0 \\ C_{12} & C_{22} & C_{23} & 0 & 0 & 0 \\ C_{13} & C_{23} & C_{33} & 0 & 0 & 0 \\ 0 & 0 & 0 & C_{44} & 0 & 0 \\ 0 & 0 & 0 & 0 & C_{55} & 0 \\ 0 & 0 & 0 & 0 & 0 & C_{66} \end{bmatrix}$

Table I.2: Independent elastic constants for an orthorhombic/orthotropic crystal[223]

Crystal:	Monoclinic
Sides:	$a \neq b \neq c$
Angles:	$\alpha < \frac{\pi}{2}$ radians, $\beta = \gamma = \frac{\pi}{2}$ radians
Symmetry:	symmetry in the xy plane
No. Independent Elastic Constants:	13
Elastic Constants:	$\begin{bmatrix} C_{11} & C_{12} & C_{13} & 0 & 0 & C_{16} \\ C_{12} & C_{22} & C_{23} & 0 & 0 & C_{26} \\ C_{13} & C_{23} & C_{33} & 0 & 0 & C_{36} \\ 0 & 0 & 0 & C_{44} & C_{45} & 0 \\ 0 & 0 & 0 & C_{45} & C_{55} & 0 \\ C_{16} & C_{26} & C_{36} & 0 & 0 & C_{66} \end{bmatrix}$

Table I.3: Independent elastic constants for a monoclinic crystal[223]

Crystal:	Triclinic
Sides:	$a \neq b \neq c$
Angles:	$\alpha < \frac{\pi}{2}$ radians, $\beta < \frac{\pi}{2}$ radians, $\gamma < \frac{\pi}{2}$ radians
Symmetry:	no symmetry
No. Independent Elastic Constants:	21
Elastic Constants:	$\begin{bmatrix} C_{11} & C_{12} & C_{13} & 0 & 0 & C_{16} \\ C_{12} & C_{22} & C_{23} & 0 & 0 & C_{26} \\ C_{13} & C_{23} & C_{33} & 0 & 0 & C_{36} \\ 0 & 0 & 0 & C_{44} & C_{45} & 0 \\ 0 & 0 & 0 & C_{45} & C_{55} & 0 \\ C_{16} & C_{26} & C_{36} & 0 & 0 & C_{66} \end{bmatrix}$

Table I.4: Independent elastic constants for a triclinic crystal[223]

I.2 Sets of Strains Applied to Calculate Elastic Constants

Strains are applied to a relaxed cell to determine the elastic constants. Various methods have been used, and these differ from paper to paper based on the crystal structure and the author. Several of these strains have been used throughout this work in various computer codes, including the QE_EoS and EAMPA codes.

I.2.1 Cubic Crystals - Mehl, Singh, Klein and Papaconstantopoulos 1993

e_1	e_2	e_3	e_4	e_5	e_6	$E(\sigma)$
σ	σ	σ	0	0	0	BM EoS Fit B_0
σ	$-\sigma$	$\frac{\sigma^2}{1-\sigma^2}$	0	0	0	$\Delta E(\sigma) = \Delta E(-\sigma) = V_0((C_{11} - C_{12})\frac{\sigma}{2} + O[\sigma^4])$
0	0	$\frac{\sigma^2}{4-\sigma^2}$	0	0	σ	$\Delta E(\sigma) = \Delta E(-\sigma) = V_0 C_{44} \frac{\sigma}{2} + O[\sigma^4]$

Table I.5: Calculation of elastic constants using the equation of state, a volume conserving orthorhombic strain and volume conserving monoclinic strain.[131][124]

These values are inserted into the strain matrix in eq. I.1.

$$\vec{\epsilon} = \begin{bmatrix} e_1 & \frac{e_6}{2} & \frac{e_5}{2} \\ \frac{e_6}{2} & e_2 & \frac{e_4}{2} \\ \frac{e_5}{2} & \frac{e_4}{2} & e_3 \end{bmatrix} \quad (\text{I.1})$$

The values for C_{11} and C_{12} are computed using the polynomial fit to the change in energy due to the second strain and the bulk modulus.

$$\begin{bmatrix} 1 & -1 \\ \frac{1}{3} & \frac{2}{3} \end{bmatrix} \begin{bmatrix} C_{11} \\ C_{12} \end{bmatrix} = \begin{bmatrix} \frac{C_p}{V_0} \\ B_0 \end{bmatrix} \quad (\text{I.2})$$

I.2.2 Orthorhombic Crystals - Ravindran, Fast, Korzhavyi and Johansson

e_1	e_2	e_3	e_4	e_5	e_6	$E(\sigma)$
σ	0	0	0	0	0	$E(V, \sigma) = E(V_0, 0) + V_0 (\tau_1 \sigma + \frac{C_{11}}{2} \sigma^2)$
0	σ	0	0	0	0	$E(V, \sigma) = E(V_0, 0) + V_0 (\tau_2 \sigma + \frac{C_{22}}{2} \sigma^2)$
0	0	σ	0	0	0	$E(V, \sigma) = E(V_0, 0) + V_0 (\tau_3 \sigma + \frac{C_{33}}{2} \sigma^2)$
$\frac{1}{(1-\sigma^2)^{\frac{1}{3}}}$	$\frac{1}{(1-\sigma^2)^{\frac{1}{3}}}$	$\frac{1}{(1-\sigma^2)^{\frac{1}{3}}}$	$\frac{1}{(1-\sigma^2)^{\frac{1}{3}}}$	0	0	$E(V, \sigma) = E(V_0, 0) + V_0 (2\tau_4 \sigma + 2C_{44} \sigma^2)$
$\frac{1}{(1-\sigma^2)^{\frac{1}{3}}}$	$\frac{1}{(1-\sigma^2)^{\frac{1}{3}}}$	$\frac{1}{(1-\sigma^2)^{\frac{1}{3}}}$	0	$\frac{1}{(1-\sigma^2)^{\frac{1}{3}}}$	0	$E(V, \sigma) = E(V_0, 0) + V_0 (2\tau_5 \sigma + 2C_{55} \sigma^2)$
$\frac{1}{(1-\sigma^2)^{\frac{1}{3}}}$	$\frac{1}{(1-\sigma^2)^{\frac{1}{3}}}$	$\frac{1}{(1-\sigma^2)^{\frac{1}{3}}}$	0	0	$\frac{1}{(1-\sigma^2)^{\frac{1}{3}}}$	$E(V, \sigma) = E(V_0, 0) + V_0 (2\tau_6 \sigma + 2C_{66} \sigma^2)$
$\frac{1+\sigma}{(1-\sigma^2)^{\frac{1}{3}}}$	$\frac{1-\sigma}{(1-\sigma^2)^{\frac{1}{3}}}$	$\frac{1}{(1-\sigma^2)^{\frac{1}{3}}}$	0	0	0	$E(V, \sigma) = E(V_0, 0) + V_0 ((\tau_1 - \tau_2) \sigma + \frac{C_{11} + C_{22} - 2C_{12}}{2} \sigma^2)$
$\frac{1+\sigma}{(1-\sigma^2)^{\frac{1}{3}}}$	$\frac{1}{(1-\sigma^2)^{\frac{1}{3}}}$	$\frac{1-\sigma}{(1-\sigma^2)^{\frac{1}{3}}}$	0	0	0	$E(V, \sigma) = E(V_0, 0) + V_0 ((\tau_1 - \tau_3) \sigma + \frac{C_{11} + C_{33} - 2C_{13}}{2} \sigma^2)$
$\frac{1}{(1-\sigma^2)^{\frac{1}{3}}}$	$\frac{1+\sigma}{(1-\sigma^2)^{\frac{1}{3}}}$	$\frac{1-\sigma}{(1-\sigma^2)^{\frac{1}{3}}}$	0	0	0	$E(V, \sigma) = E(V_0, 0) + V_0 ((\tau_2 - \tau_3) \sigma + \frac{C_{22} + C_{33} - 2C_{23}}{2} \sigma^2)$

Table I.6: Calculation of orthorhombic elastic constants by Ravindran et al.[dfttisiravindran]

These values are inserted into the strain matrix in eq. I.3.

$$\vec{\epsilon} = \begin{bmatrix} e_1 & e_6 & e_5 \\ e_6 & e_2 & e_4 \\ e_5 & e_4 & e_3 \end{bmatrix} \quad (\text{I.3})$$

The 9 elastic constants are calculated by polynomial fit to the strain-energy data. Using the second order of each polynomial, p_1 to p_9 , the elastic constants are calculated by solving eq I.4.

$$\begin{bmatrix} \frac{V_0}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{V_0}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{V_0}{2} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2V_0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2V_0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 2V_0 & 0 & 0 & 0 \\ \frac{V_0}{2} & \frac{V_0}{2} & 0 & 0 & 0 & 0 & -V_0 & 0 & 0 \\ \frac{V_0}{2} & 0 & \frac{V_0}{2} & 0 & 0 & 0 & 0 & -V_0 & 0 \\ 0 & \frac{V_0}{2} & \frac{V_0}{2} & 0 & 0 & 0 & 0 & 0 & -V_0 \end{bmatrix} \begin{bmatrix} C_{11} \\ C_{22} \\ C_{33} \\ C_{44} \\ C_{55} \\ C_{66} \\ C_{12} \\ C_{13} \\ C_{23} \end{bmatrix} = \begin{bmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \\ p_5 \\ p_6 \\ p_7 \\ p_8 \\ p_9 \end{bmatrix} \quad (I.4)$$

I.2.3 Cubic Crystals - Connetable and Thomas 2009

e_1	e_2	e_3	e_4	e_5	e_6	$E(\sigma)$
σ	σ	σ	0	0	0	BM EoS Fit B_0
σ	0	0	0	0	0	$\Delta E(\sigma) = \frac{V_0}{2}C_{11}\sigma^2 + O[\sigma^4]$
0	0	0	2σ	2σ	2σ	$\Delta E(\sigma) = 6V_0C_{44}\sigma^2 + O[\sigma^4]$

Table I.7: Calculation of elastic constants for a cubic crystal[155]

$$\vec{\epsilon} = \begin{bmatrix} 2e_1 & e_6 & e_5 \\ e_6 & 2e_2 & e_4 \\ e_5 & e_4 & 2e_3 \end{bmatrix} \quad (I.5)$$

I.2.4 Orthorhombic Crystals - Connetable and Thomas 2009

$$\vec{\epsilon} = \begin{bmatrix} 2e_1 & e_6 & e_5 \\ e_6 & 2e_2 & e_4 \\ e_5 & e_4 & 2e_3 \end{bmatrix} \quad (I.6)$$

I.2.5 Stability Conditions

Whilst the elastic constants may be computed using first-principles calculations, there are a series of assumptions and approximations made in order to solve first-principles calculations in a reasonable amount of time. There

e_1	e_2	e_3	e_4	e_5	e_6	$E(\sigma)$
σ	0	0	0	0	0	$\Delta E(\sigma) = \frac{V_0}{2} C_{11} \sigma^2$
0	σ	0	0	0	0	$\Delta E(\sigma) = \frac{V_0}{2} C_{22} \sigma^2$
0	0	σ	0	0	0	$\Delta E(\sigma) = \frac{V_0}{2} C_{33} \sigma^2$
0	0	0	2σ	0	0	$\Delta E(\sigma) = 2V_0 C_{44} \sigma^2$
0	0	0	0	2σ	0	$\Delta E(\sigma) = 2V_0 C_{55} \sigma^2$
0	0	0	0	0	2σ	$\Delta E(\sigma) = 2V_0 C_{66} \sigma^2$
$u(1 + \sigma) - 1$	$u(1 - \sigma) - 1$	$u - 1$	0	0	0	$\Delta E(\sigma) = \frac{V_0}{2} (C_{11} + C_{22} - 2C_{12}) \sigma^2$
$u(1 + \sigma) - 1$	$u - 1$	$u(1 - \sigma) - 1$	0	0	0	$\Delta E(\sigma) = \frac{V_0}{2} (C_{11} + C_{33} - 2C_{13}) \sigma^2$
$u - 1$	$u(1 + \sigma) - 1$	$u(1 - \sigma) - 1$	0	0	0	$\Delta E(\sigma) = \frac{V_0}{2} (C_{22} + C_{33} - 2C_{23}) \sigma^2$

Table I.8: Calculation of elastic constants for a cubic crystal[155], $u = (1 - \sigma^2)^{-\frac{1}{3}}$

must be a sanity check, as it is no use to compute the elastic constants from DFT to find the crystals are unstable.

For a cubic crystal with three independent elastic constants and zero stress, the Born elastic stability criteria apply[224]. The bulk modulus should be positive $(C_{11} + 2C_{12})/3 > 0$, the tetragonal shear should be positive $C_{11} - C_{12} > 0$ and the shear modulus should be positive $c_{44} > 0$.

For an orthorhombic crystal, there are additional conditions given there are nine independent elastic constants (eq. I.7)[225].

$$\begin{aligned} C_{11} &> 0, C_{22} > 0, C_{33} > 0, C_{44} > 0, C_{55} > 0, C_{66} > 0 \\ (C_{11} + C_{22} - 2C_{12}) &> 0, (C_{11} + C_{33} - 2C_{13}) > 0, (C_{22} + C_{33} - 2C_{23}) > 0 \\ (C_{11} + C_{22} + C_{33} + 2C_{12} + 2C_{13} + 2C_{23}) &> 0 \end{aligned} \quad (\text{I.7})$$

I.3 Computing Values from Elastic Constants

The bulk modulus, as noted earlier, is a measure of the effect of strain on stress. While it may be calculated by taking the second derivative of energy with respect to volume, or by fitting an equation of state, it may also be calculated using the elastic constants of a material or the compliance constants.

For cubic crystals the elastic constants may be used to calculate the bulk modulus, tetragonal shear, shear modulus and Cauchy pressure (eq. I.8).

$$\begin{aligned} B_0 &= \frac{C_{11} + 2C_{12}}{3} = \frac{5C_{11}}{9} \text{ Bulk Modulus} \\ C' &= \frac{C_{11} - C_{12}}{2} \text{ Tetragonal Shear} \\ &\quad C_{44} \text{ Shear Modulus} \\ p_c &= C_{12} - C_{44} \text{ Cauchy Pressure} \end{aligned} \quad (\text{I.8})$$

For an orthorhombic crystal, the Voigt bulk modulus (eq. I.9) and Reuss bulk modulus (eq. I.10) may be

calculated from the stiffness and compliance matrix respectively. These two values are then averaged to give the bulk modulus (eq. I.11)[130].

$$B_V = \frac{1}{9} (C_{11} + C_{22} + C_{33} + 2(C_{12} + C_{13} + C_{23})) \quad (\text{I.9})$$

$$B_R = (S_{11} + S_{22} + S_{33} + 2(S_{12} + S_{13} + S_{23}))^{-1} \quad (\text{I.10})$$

$$B = 0.5(B_V + B_R) \quad (\text{I.11})$$

Calculation by this method is particularly useful for this work, where the FCC iron crystal is orthorhombic, and not cubic (section 7.3.4).

Elastic constants are also used to calculate the shear modulus G . Where the crystal is cubic, there are several elastic constants that can be used (eq. I.12)[130]. Both the Voigt (eq. I.13) and Reuss (eq. I.13) values are calculated and averaged to give the value of G (eq. I.15).

$$G = C_{44} = 0.5(C_{11} - C_{12}) = \frac{C_{11}}{3} \quad (\text{I.12})$$

$$G_V = \frac{1}{15} (C_{11} + C_{22} + C_{33} - C_{12} - C_{13} - C_{23}) + \frac{1}{5} (C_{44} + C_{44} + C_{55}) \quad (\text{I.13})$$

$$G_R = \frac{15}{4(S_{11} + S_{22} + S_{33}) - 4(S_{12} + S_{23} + S_{32}) + 3(S_{44} + S_{55} + S_{66})} \quad (\text{I.14})$$

$$G = 0.5(G_V + G_R) \quad (\text{I.15})$$

The Young's modulus E as a scalar is a measure of how easily the length an isotropic material is changed under tension or compression, and the computed elastic constants for a cubic crystal may be used to calculate E (eq. I.16).

$$E = \frac{9B_0G}{3B_0 + G} \text{ where } B_0 \text{ is the bulk modulus and } G \text{ is the shear modulus} \quad (\text{I.16})$$

The Poisson ration is a measure of how much a material changes in the direction perpendicular to a strain applied transversely.

$$\nu = -\frac{d\epsilon_{perp}}{d\epsilon_{transverse}} \quad (I.17)$$

The bulk modulus may be calculated from the equation of state or elastic constants, and the shear modulus may be calculated from the elastic constants. In turn, the Poisson ratio is calculated from these.

$$\nu = \frac{3B_0 - 2G}{2(3B + G)} \quad (I.18)$$

The ratio may be used as another data point in order to fit a potential, or at the very least as a check to compare to the known value once a potential has been derived.

I.4 Correlation of Melting Temperature in Metals with Elastic Constants

There is a correlation between the melting temperature of metals and their elastic constants detailed in eq. I.19 and fig. I.1) [226].

$$T \approx 598.0 + 6.66 \times (C_{11} + C_{22} + C_{33}) - 0.003 \times (C_{11} + C_{22} + C_{33}) \quad (I.19)$$

Where the temperature is in K, and the elastic constants are in GPA

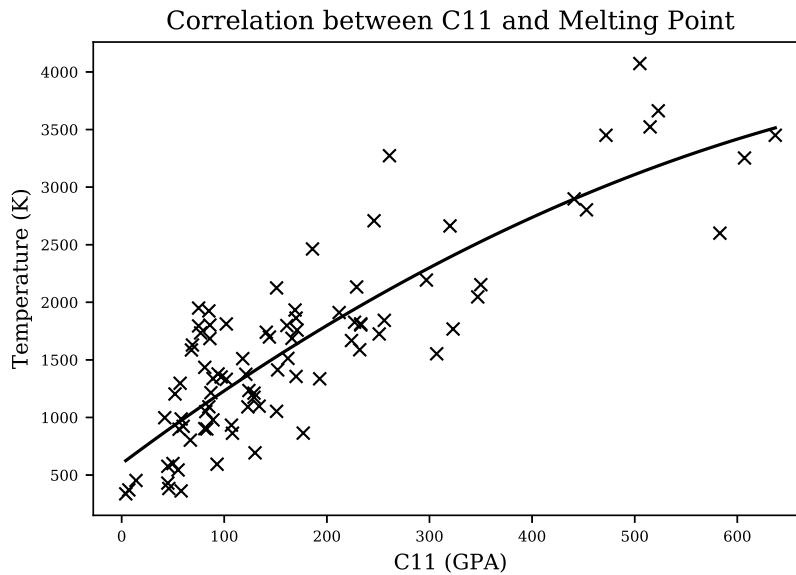


Figure I.1: Correlation between C_{11} and temperature

The correlation values between the temperature and C_{11} are 0.848 and 0.780 for the Pearsons and Spearmans correlation respectively. An accurate temperature will not be possible to predict, but it will act as a further sanity check within the computer codes developed in the results section of this work.

Appendix J

PBE Functional

The GGA PBE functional was the primary type used in this work for the DFT calculated data. The LDA, LSDA and GGA are described in detail in “Density functionals from LDA to GGA”[168].

The local density approximation is:

$$E_{xc}^{LDA}[\rho] = \int d^3r \rho(\vec{r}) [e_x(\rho(\vec{r})) + e_c(\rho(\vec{r}))] \quad (\text{J.1})$$

Exchange energy per electron of the unpolarized uniform electron gas:

$$e_x(\rho) \quad (\text{J.2})$$

Correlation energy per electron of the unpolarized uniform electron gas:

$$e_c(\rho) \quad (\text{J.3})$$

Splitting the density into spin up and spin down:

$$\begin{aligned} \rho_{\uparrow}(\vec{r}) &= \sum_k^{occ} |\phi_{k,\uparrow}(\vec{r})|^2 \\ \rho_{\downarrow}(\vec{r}) &= \sum_k^{occ} |\phi_{k,\downarrow}(\vec{r})|^2 \\ \rho(\vec{r}) &= \rho_{\uparrow}(\vec{r}) + \rho_{\downarrow}(\vec{r}) \end{aligned} \quad (\text{J.4})$$

The relative spin polarization is:

$$\zeta = \frac{\rho_{\uparrow} - \rho_{\downarrow}}{\rho_{\uparrow} + \rho_{\downarrow}} \quad (\text{J.5})$$

The LSDA is:

$$E_{xc}^{LSDA}[\rho_\uparrow, \rho_\downarrow] = \int d^3r \rho(\vec{r}) [e_x(\rho(\vec{r})) f(\zeta(\vec{r})) + e_c(r_s(\vec{r}), \zeta(\vec{r}))] \quad (\text{J.6})$$

where

$$f(\zeta) = \frac{1}{2} \left[(1 + \zeta)^{\frac{4}{3}} + (1 - \zeta)^{\frac{4}{3}} \right] \quad (\text{J.7})$$

The correlation energy per electron of the spin-polarized uniform electron gas is:

$$e_c(r_s, \zeta) \quad (\text{J.8})$$

where r_s is the local Seitz radius:

$$\begin{aligned} \rho &= 3/4\pi r_s^3 \\ r_s &= \sqrt[3]{\frac{4\rho}{3\pi}} \end{aligned} \quad (\text{J.9})$$

The PBE correlation energy functional is:

$$E_c^{PBE}[\rho_\uparrow, \rho_\downarrow] = \int d^3r \rho [e_c(r_s, \zeta) + H(r_s, \zeta, t)] \quad (\text{J.10})$$

where

$$t = \left(\frac{\pi}{4}\right)^{\frac{1}{2}} \left(\frac{9\pi}{4}\right)^{\frac{1}{6}} \frac{s}{\phi r_s^{\frac{1}{2}}} \quad (\text{J.11})$$

$$s = \frac{|\nabla \rho|}{2(3\pi^2)^{\frac{1}{3}} \rho^{\frac{4}{3}}} = \frac{3}{2} \left(\frac{4}{9\pi}\right)^{\frac{1}{3}} a_B |\nabla r_s| \quad (\text{J.12})$$

$$a_B = \frac{\hbar^2}{m\epsilon^2} \text{ (the Bohr radius)} \quad (\text{J.13})$$

$$\phi(\zeta) = \frac{1}{2} \left[(1 + \zeta)^{\frac{2}{3}} + (1 - \zeta)^{\frac{2}{3}} \right] \quad (\text{J.14})$$

$$H = \gamma \phi^3 \ln \left[1 + \frac{\beta}{\gamma} t^2 \left(\frac{1 + At^2}{1 + At^2 + A^2 t^4} \right) \right] \quad (\text{J.15})$$

$$A = \left(\frac{\beta}{\gamma} \right) \frac{1}{\exp(-e_c(r_s, \zeta)/(\gamma\phi^3)) - 1} \quad (\text{J.16})$$

where $\beta = 0.066725$ and $\gamma = 0.031091$ Hartree.

The PBE exchange energy functional is:

$$E_x^{PBE}[\rho] = \int d^3r \rho e_x(\rho) F_x(s) \quad (\text{J.17})$$

where

$$F_x(s) = 1 + \kappa - \frac{\kappa}{1 + \mu s^2/\kappa} \quad (\text{J.18})$$

where $\kappa = 0.804$ and $\mu = 0.21951$.

The spin density functional is:

$$E_x^{PBE}[\rho_\uparrow, \rho_\downarrow] = \frac{1}{2} E_x^{PBE}[2\rho_\uparrow] + \frac{1}{2} E_x^{PBE}[2\rho_\downarrow] \quad (\text{J.19})$$

Appendix K

QECONVERGE

The full code is available at <https://github.com/BenPalmer1983/qeconverge>

Listing K.1: QECONVERGE Readme File

QECONVERGE

Python code.

Before running DFT calculations, the ecutwfc and ecutrho values need to be chosen. They are minimised such that the values are within a given converged limit. They are also minimised because using an arbitrarily large cutoff would affect computer resources and computational time.

A suitable k-point mesh and smearing must also be determined.

Qeconverge takes an input file and a template pwscf file. The position of the atoms can be varied slightly. It starts at a specified ecutwfc and uses ecutrho = 4.0 x ecutwfc. The value of ecutwfc (and ecutrho) are increased until the energy and forces converge. The next stage is to reduce ecutrho while remaining in the convergence threshold (fixing ecutwfc). Finally ecutrho is fixed and the code attempts to reduce ecutwfc further.

Further calculations are run and a 2D map ecutwfc on one axis and ecutrho on the other is mapped to visualise the convergence for the user.

Finally a 2D map with k-points on one axis and smearing on the other is created, to aid the user in choosing a k-point and smearing values.

Example 1:

examples/example1

The run file packages the python files into one called qeconverge.py - it then runs the python program. System settings are stored in environment variables in the run.sh bash file:

```
export OMP_NUM_THREADS=1
export PROC_COUNT=4
export PWSCF_SCRATCH=/opt/scratch
export PWSCF_PP=/opt/pp
export PWSCF_CACHE=/opt/pwscf_cache
export PWSCF_BIN=/opt/qe/bin/pw.x
```

Example 2:

```
examples/example2
```

Same as example 1, but it runs the python file in its own directory.

Example 3:

```
examples/example3
```

This is similar to example 2, but the run.sh file is replaced with a batch.sh file - the names don't matter, but the batch.sh file contains more settings that are used on the BlueBEAR computer. It is submitted to the job queue:

```
sbatch batch.sh
```

The settings may be modified to work on other supercomputers with different scheduling programs. The python module and quantum espresso modules may need to be loaded, although I use my own compiled quantum espresso.

batch.sh:

```
#!/bin/bash
#
#SBATCH --job-name=pwaleos
#SBATCH --output=jobout.txt
#SBATCH --account=readmsd02
#
#SBATCH --ntasks 20
#SBATCH --nodes 1
#SBATCH --time=360:00
#SBATCH --cpus-per-task=1
#
#SBATCH --get-user-env
#SBATCH --export=NONE
#
unset SLURM_EXPORT_ENV

module purge; module load bluebear
module load bear-apps/2018a
module load iomkl/2018a
module load Python/3.6.3-iomkl-2018a
module load matplotlib/2.1.1-iomkl-2018a-Python-3.6.3

# Change to $PBS_O_WORKDIR
cd "$PBS_O_WORKDIR"
# Set the number of threads to 1
export OMP_NUM_THREADS=1
export PROC_COUNT=20
export PWSCF_SCRATCH=/scratch
export PWSCF_PP=/rds/homes/b/bxp912/pp
export PWSCF_CACHE=/rds/homes/b/bxp912/pwscf_cache
export PWSCF_BIN=/rds/homes/b/bxp912/apps/qe-6.3/bin/pw.x

python qeconverge.py input.in > result.txt
```

Appendix L

DFT Convergence Plots

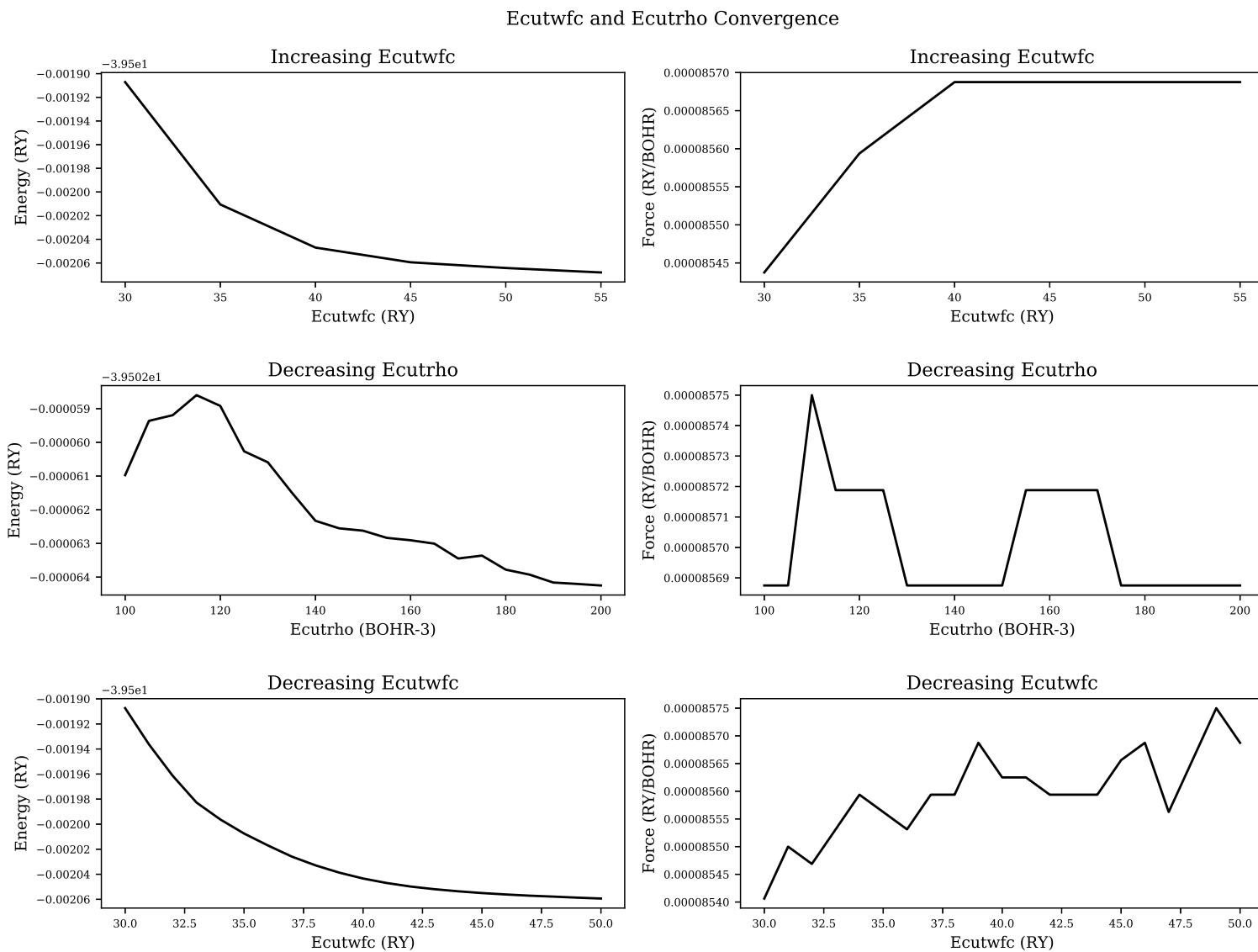


Figure L.1: Ecutwfc and Ecutfro convergence for aluminium (energy and force)

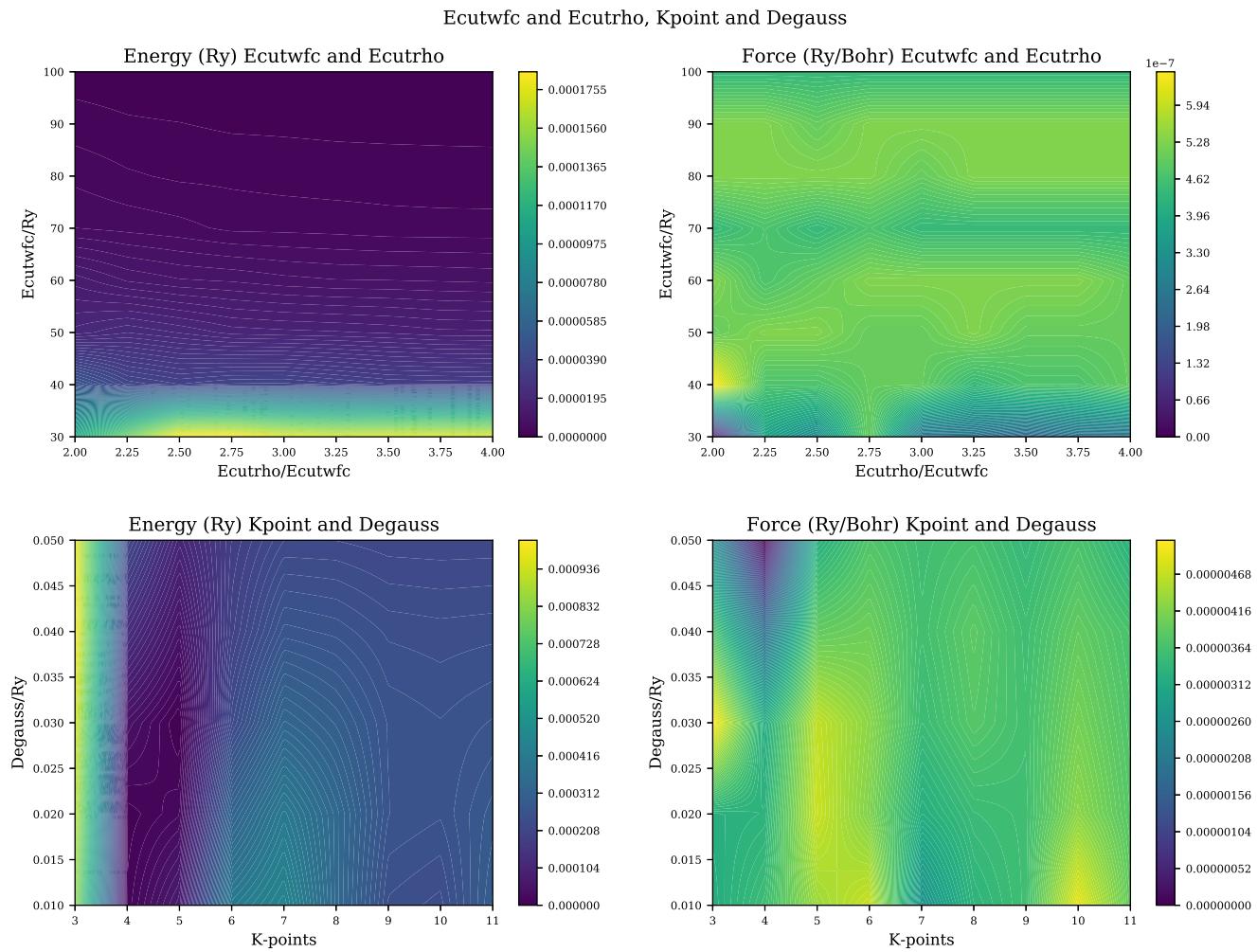


Figure L.2: 2D density map - force and energy convergence with k-points and smearing - aluminium

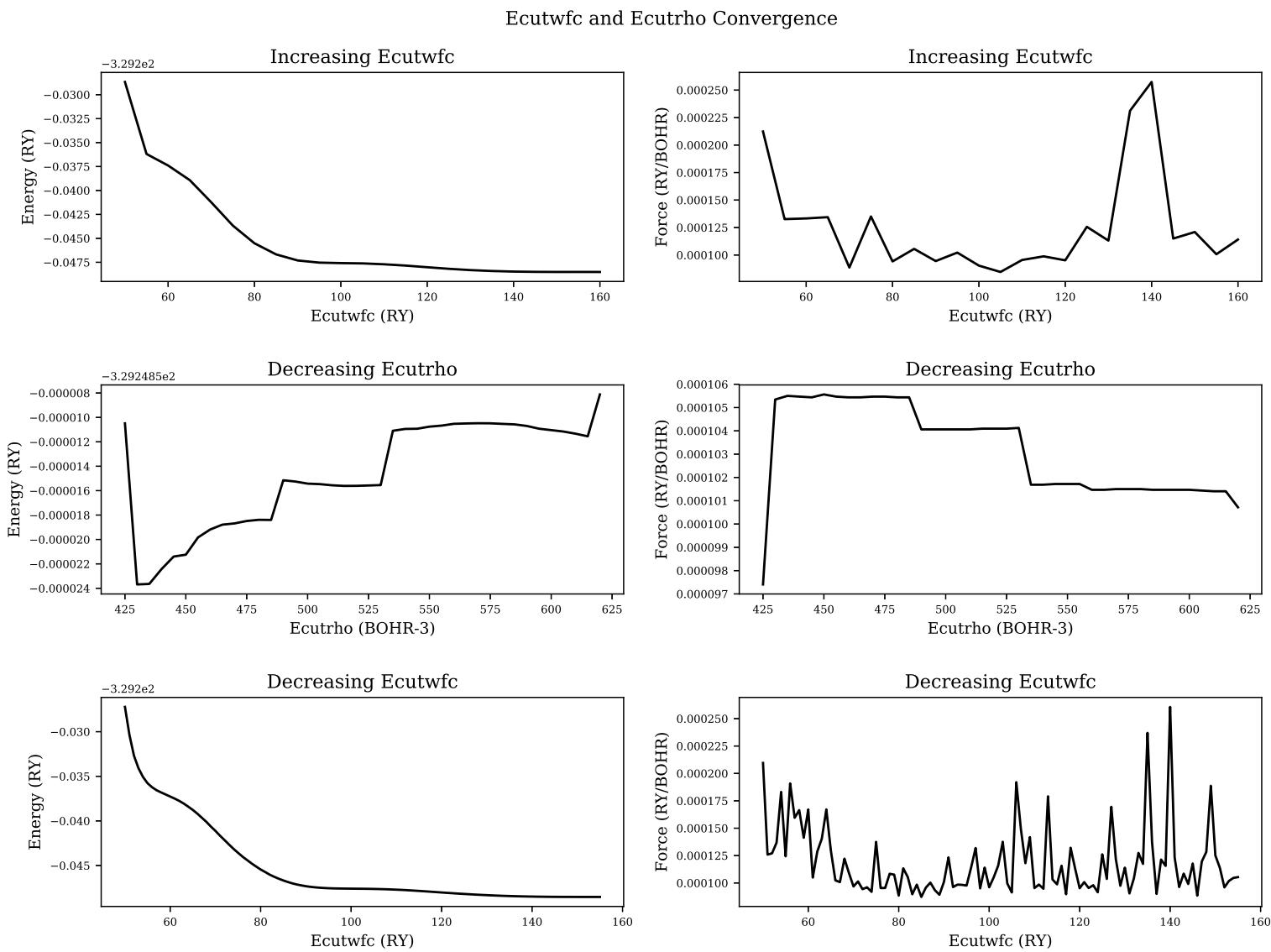


Figure L.3: Ecutwfc and Ecutrho convergence for iron (energy and force)

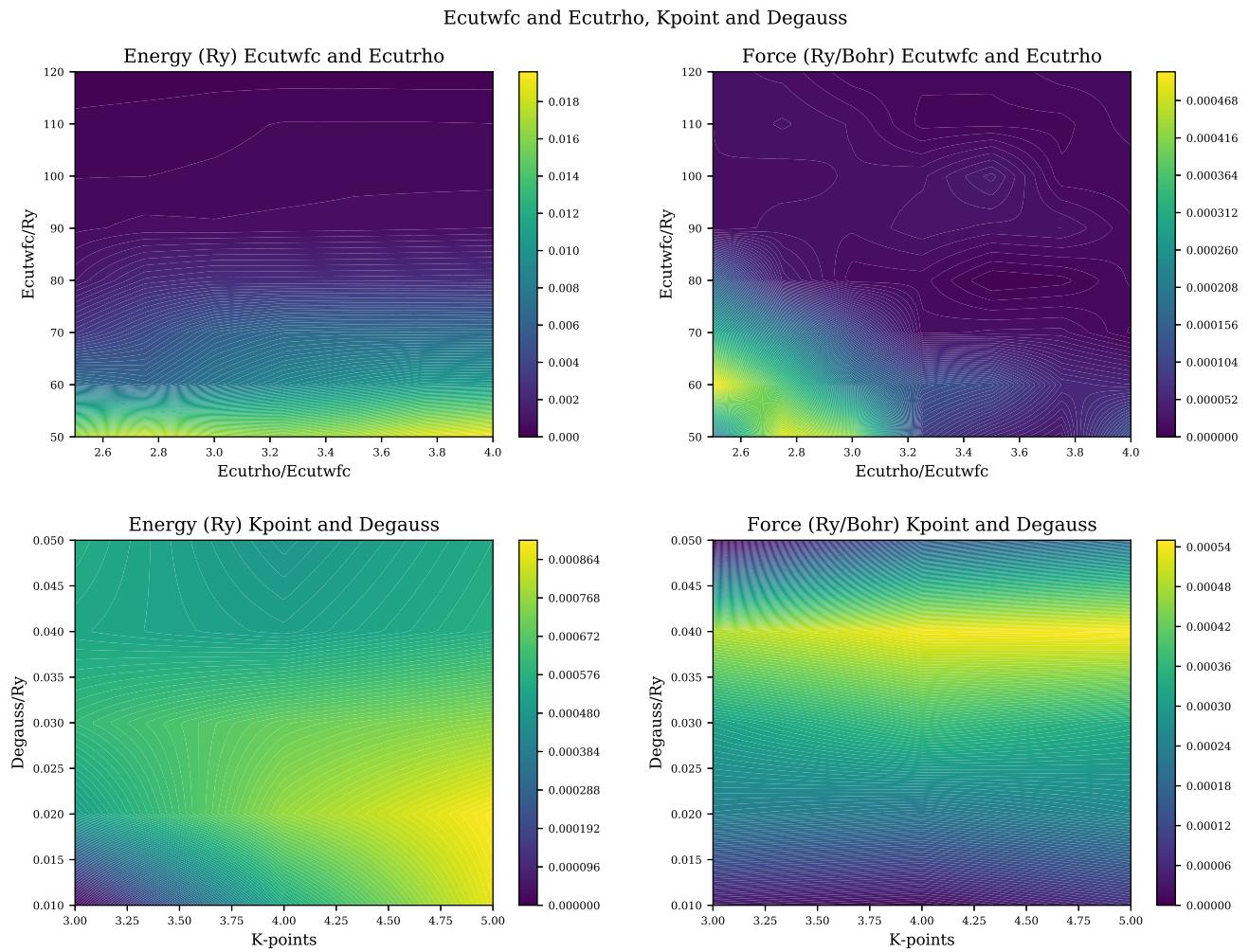


Figure L.4: 2D density map - force and energy convergence with k-points and smearing - iron

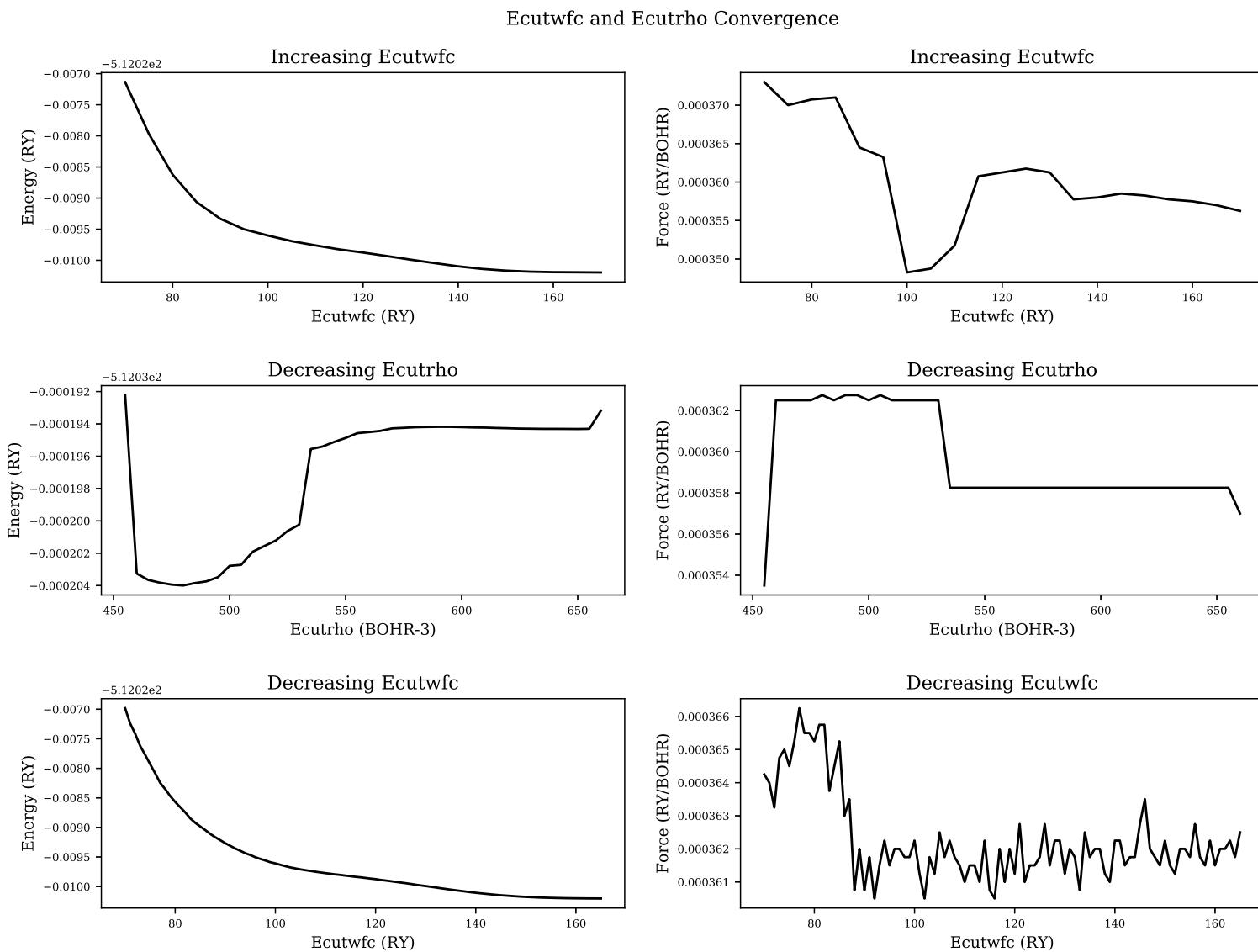


Figure L.5: Ecutwfc and Ecotrho convergence for palladium (energy and force)

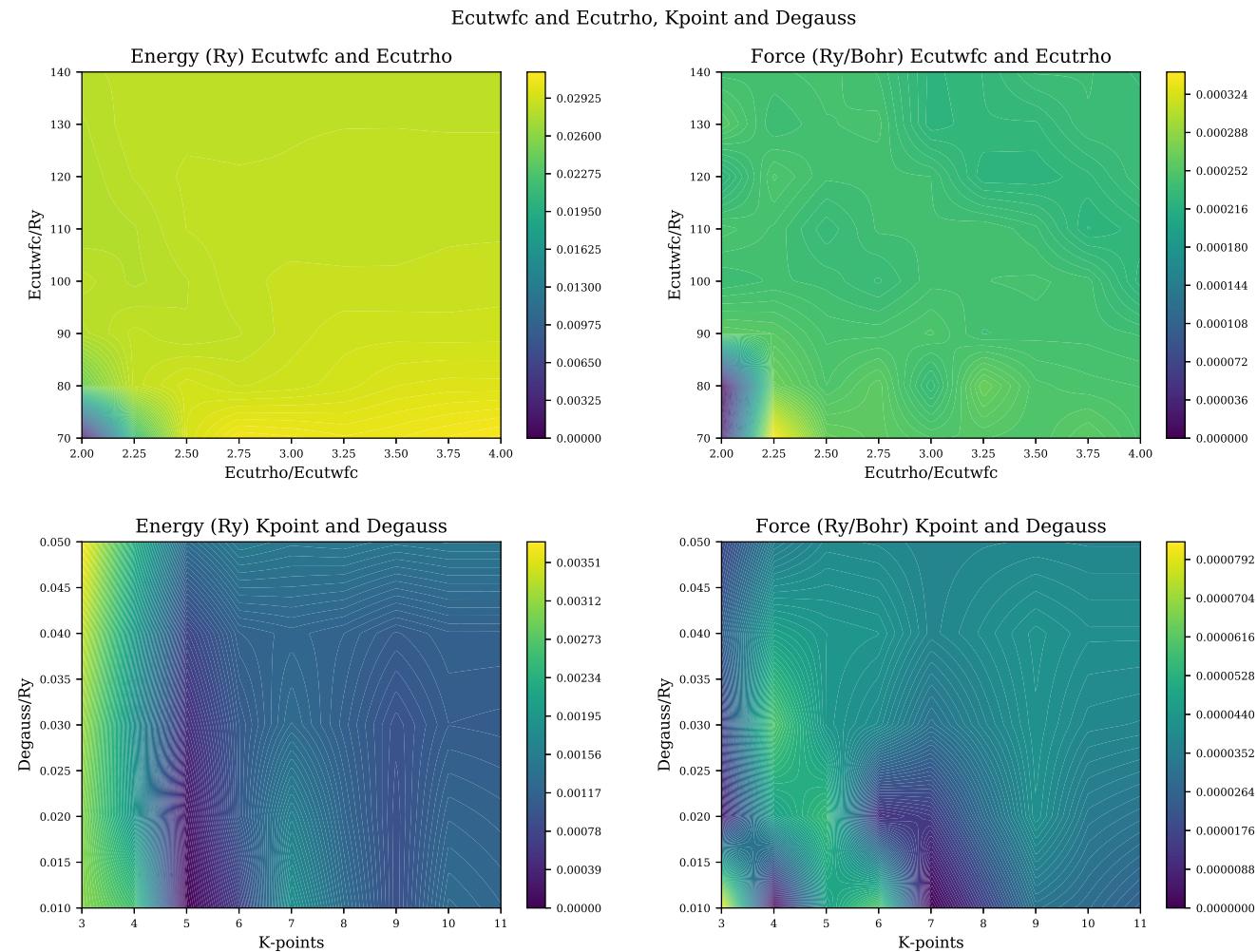


Figure L.6: 2D density map - force and energy convergence with k-points and smearing - palladium

Appendix M

QEEOS

The full code is available at <https://github.com/BenPalmer1983/qeeos>

Several of the important classes that make up the code are listed below.

Listing M.1: QEEOS Readme File

```
QEEOS
=====

```

```
Quantum Espresso - Equation of State (and Elastic Constants)
```

```
Python code.
```

```
This code uses pwscf from the Quantum Espresso suite to calculate crystal energies using DFT.
```

```
For a full run, this is what the code does:
```

1. Reads in the user's input
2. Creates a template file for PWscf based on the user's input and supplied template file
3. Creates a vc-relax input file and runs this to find the relaxed a0 and cell parameters
Also calculates the density of the material based on the DFT results
4. Creates and executes a series of SFC or RELAX input files over a range of strains
in order to calculate the equation of state
5. Creates and executes a series of SFC or RELAX input files over a range of strains from MSKP
in order to calculate the cubic elastic constants
6. Creates and executes a series of SFC or RELAX input files over a range of strains from RFKJ
in order to calculate the orthorhombic elastic constants
7. Processes the results
8. Creates plots
9. Using the elastic constants, other material properties are calculated or estimated

Melh, Singh, Klein, Papaconstatopoulos
Ravindran, Fast, Korzhavyi, Johansson

Listing M.2: Calculation of Bulk Property

```
1 #!/bin/python3
2
3 import os
4 import sys
5 import numpy
6 import time
7 import shutil
8
```

```

9  from g import g
10 from ds import ds
11
12 from display import display
13 from output import output
14 from units import units
15 from template import template
16 from pwscf_inout import pwscf_input
17 from pwscf_inout import pwscf_output
18 from pwout_read import pwout_read
19 from pwe import pwe
20 from transforms import transforms
21 from adjust_energy import adjust_energy
22
23
24
25 class eos_bm:
26
27
28     @staticmethod
29     def run():
30         if(not g.isfile['eos_bm']):
31             return False
32         output.log("Run PWscf for EoS BM", verbose=0)
33
34         # Get total strain and steps
35         eos_strain = g.isfile['eos_bm_strain']
36         eos_steps = g.isfile['eos_bm_steps']
37
38         # Make
39         g.results['eos_bm_se'] = numpy.zeros((2 * eos_steps + 1, 2,), dtype=numpy.float64)
40         g.results['eos_bm_ve'] = numpy.zeros((2 * eos_steps + 1, 2,), dtype=numpy.float64)
41
42         # Load template
43         eos = template.load_relaxed()
44         uv = eos.get_cp_array()
45
46         # Prepare files
47         files = []
48         n = 0
49         for i in range(-eos_steps, eos_steps+1):
50             n_str = str(n + 1)
51             while(len(n_str)<4):
52                 n_str = "0" + n_str
53
54             # Make and store strain
55             s = eos_strain * (i / eos_steps)
56             g.results['eos_bm_se'][n, 0] = s
57
58             # Transform
59             d = transforms.eos(s)
60             uv_new = numpy.matmul(d, uv)
61
62             # Update pw obj
63             eos.set_prefix()
64             eos.set_cp_arr(uv_new)
65             if(g.isfile['eos_bm_calc'] == 'relax'):
66                 eos.set_calculation("relax")
67             else:
68                 eos.set_calculation("scf")
69             eos.save("eos_bm_" + n_str + ".in", g.dirs['eos_bm'])

```

```
70     files.append(os.path.join(g.dirs['eos_bm'], "eos_bm_" + n_str + ".in"))
71
72     # Increment
73     n = n + 1
74
75
76
77     # Run PWscf
78     output.log("Running PWscf", verbose=0)
79     out = pwe.run(files)
80
81
82     output.log("Read EoS BM results", verbose=0)
83     # Read results
84     for n in range(len(out)):
85         o = pwout_read.read(out[n][1])
86         vol = units.convert('BOHR3', 'ANG3', o['vpa'])
87         energy = adjust_energy.run(o['energy'], o['labels'], unit_in='ry', epa=True)
88
89         g.results['eos_bm_se'][n, 1] = energy
90         g.results['eos_bm_ve'][n, 0] = vol
91         g.results['eos_bm_ve'][n, 1] = energy
92
93     # Save to file
94     output.log("Save EoS BM results to file", verbose=0)
95     numpy.savetxt(g.files['eos_bm'], g.results['eos_bm_ve'], delimiter=",")
```

Listing M.3: Calculation of Elastic Constants - MSKP

```
1 #!/bin/python3
2
3 import os
4 import sys
5 import numpy
6 import time
7 import shutil
8
9 from g import g
10 from ds import ds
11
12 from display import display
13 from output import output
14 from units import units
15 from template import template
16 from pwscf_inout import pwscf_input
17 from pwscf_inout import pwscf_output
18 from pwout_read import pwout_read
19 from pwe import pwe
20 from transforms import transforms
21 from adjust_energy import adjust_energy
22
23
24
25
26 class ec_mskp:
27
28     @staticmethod
29     def run():
30         if(not g.isfile['ec_mskp']):
31             return False
32         output.log("Run PWscf for EC MSKP", verbose=0)
```

```

34
35     # Get total strain and steps
36     ec_mskp_strain = g.isfile['ec_mskp_strain']
37     ec_mskp_steps = g.isfile['ec_mskp_steps']
38
39     # Make
40     g.results['ec_mskp_se'] = {}
41
42     # Load template
43     ec = template.load_relaxed()
44     uv = ec.get_cp_array()
45
46     # Prepare files
47     files = []
48
49     for stype in g.mskp:
50         g.results['ec_mskp_se'][stype] = numpy.zeros((ec_mskp_steps, 2,), dtype=numpy.float64)
51         n = 0
52         for i in range(ec_mskp_steps):
53             n_str = str(n + 1)
54             while(len(n_str)<4):
55                 n_str = "0" + n_str
56
57             # Make and store strain
58             s = ec_mskp_strain * (i / ec_mskp_steps)
59             g.results['ec_mskp_se'][stype][n, 0] = s
60
61             # Transform
62             d = transforms.ec_mskp(s, stype)
63             uv_new = numpy.matmul(d, uv)
64
65             # Update pw obj
66             ec.set_prefix()
67             ec.set_cp_arr(uv_new)
68             if(g.isfile['ec_mskp_calc'] == 'relax'):
69                 ec.set_calculation("relax")
70             else:
71                 ec.set_calculation("scf")
72             ec.save("ec_mskp_" + stype + "_" + n_str + ".in", g.dirs['ec_mskp'])
73             files.append(os.path.join(g.dirs['ec_mskp'], "ec_mskp_" + stype + "_" + n_str + ".in"))
74
75             # Increment
76             n = n + 1
77
78     # Run PWscf
79     output.log("Running PWscf", verbose=0)
80     out = pwe.run(files)
81
82     # Read results
83     output.log("Read EC MSKP results", verbose=0)
84     n = 0
85     for stype in g.mskp:
86         m = 0
87         for i in range(ec_mskp_steps):
88             o = pwout_read.read(out[n][1])
89             vol = units.convert('BOHR3', 'ANG3', o['vpa'])
90             energy = adjust_energy.run(o['energy'], o['labels'], unit_in='ry', epa=True)
91             g.results['ec_mskp_se'][stype][m, 1] = energy
92             m = m + 1
93         n = n + 1
94

```

```
95     # Save to file
96     output.log("Save EC MSKP results to file", verbose=0)
97     for stype in g.mskp:
98         numpy.savetxt(g.files['ec_mskp'][stype], g.results['ec_mskp_se'][stype], delimiter=",")
```

Listing M.4: Calculation of Elastic Constants - RFKJ

```
1 #!/bin/python3
2
3 import os
4 import sys
5 import numpy
6 import time
7 import shutil
8
9 from g import g
10 from ds import ds
11
12 from display import display
13 from output import output
14 from units import units
15 from template import template
16 from pwscf_inout import pwscf_input
17 from pwscf_inout import pwscf_output
18 from pwout_read import pwout_read
19 from pwe import pwe
20 from transforms import transforms
21 from adjust_energy import adjust_energy
22
23
24
25
26 class ec_rfkj:
27
28     @staticmethod
29     def run():
30         if(not g.isfile['ec_rfkj']):
31             return False
32         output.log("Run PWscf for EC RFKJ", verbose=0)
33
34         # Get total strain and steps
35         strain = g.isfile['ec_rfkj_strain']
36         steps = g.isfile['ec_rfkj_steps']
37
38         # Make
39         g.results['ec_rfkj_se'] = {}
40
41         # Load template
42         ec = template.load_relaxed()
43         uv = ec.get_cp_array()
44
45         # Prepare files
46         files = []
47
48         for stype in g.rfkj:
49             g.results['ec_rfkj_se'][stype] = numpy.zeros((2*steps+1, 2,), dtype=numpy.float64)
50             n = 0
51             #for i in range(steps):
52             for i in range(-steps, steps+1):
53                 n_str = str(n + 1)
54                 while(len(n_str)<4):
```

```

56     n_str = "0" + n_str
57
58     # Make and store strain
59     s = strain * (i / steps)
60     g.results['ec_rfkj_se'][stype][n, 0] = s
61
62     # Transform
63     d = transforms.ec_rfkj(s, stype)
64     uv_new = numpy.matmul(d, uv)
65
66     # Update pw obj
67     ec.set_prefix()
68     ec.set_cp_arr(uv_new)
69     if(g.ifile['ec_rfkj_calc'] == 'relax'):
70         ec.set_calculation("relax")
71     else:
72         ec.set_calculation("scf")
73     ec.save("ec_rfkj_" + stype + "_" + n_str + ".in", g.dirs['ec_rfkj'])
74     files.append(os.path.join(g.dirs['ec_rfkj'], "ec_rfkj_" + stype + "_" + n_str + ".in"))
75
76     # Increment
77     n = n + 1
78
79     # Run Pwscf
80     output.log("Running Pwscf", verbose=0)
81     out = pwe.run(files)
82
83     # Read results
84     output.log("Read EC RFKJ results", verbose=0)
85     for stype in g.rfkj:
86         n = 0
87         for i in range(-steps, steps+1):
88             o = pwout_read.read(out[n][1])
89             vol = units.convert('BOHR3', 'ANG3', o['vpa'])
90             energy = adjust_energy.run(o['energy'], o['labels'], unit_in='ry', epa=True)
91             g.results['ec_rfkj_se'][stype][n, 1] = energy
92             n = n + 1
93
94     # Save to file
95     output.log("Save EC RFKJ results to file", verbose=0)
96     for stype in g.rfkj:
97         numpy.savetxt(g.files['ec_rfkj'][stype], g.results['ec_rfkj_se'][stype], delimiter=",")

```

Appendix N

DFT Calculated Properties

N.1 QEEOS

These results were calculated using the QEEOS python program that automatically creates and runs PWscf input files, collects the output files and processes the data.

N.2 FCC Aluminium

N.2.1 DFT Settings for PWscf

The calculations were performed with the Quantum Espresso PWscf package along with the QEEOS python program.

Setting	Value
ecutwfc (Ry)	50
ecutrho	200
smearing (Ry)	0.04
k-points	11 11 11 1 1 1
nspin	0 (Non-magnetic)
pseudopotential	Al.pbe-nl-kjpaw-psl.1.0.0.UPF
etot_conv_thr	0.0001
forc_conv_thr	0.001
conv_thr	1.0D-6
diagonalization	david
mixing_beta	0.1
mixing_mode	plain

Table N.1: Aluminium DFT settings

N.2.2 Resulting Plots

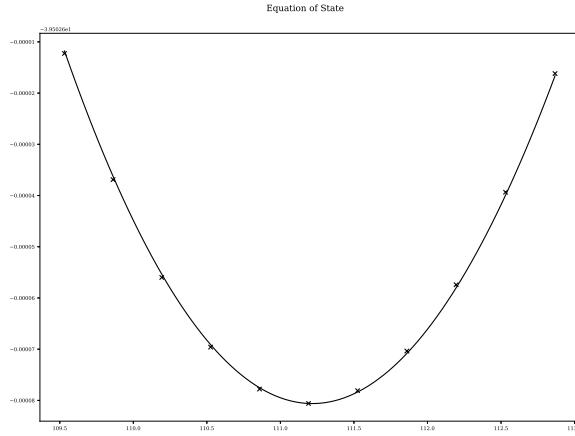


Figure N.1: Aluminium FCC equation of state plot

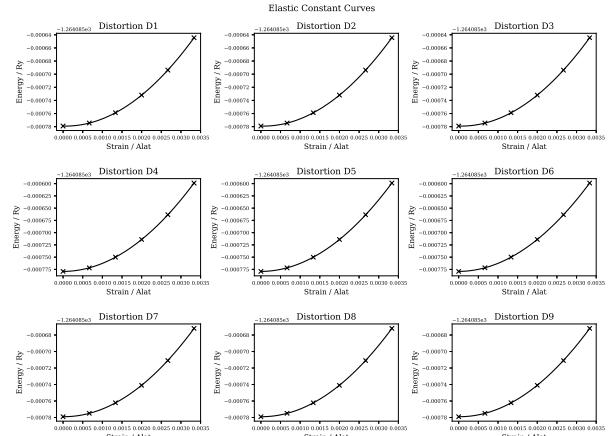


Figure N.2: Aluminium FCC elastic constants plot

N.2.3 Aluminium FCC Results File

Listing N.1: Aluminium FCC DFT Calculated Properties

```

1 ######
2 #          RESULTS          #
3 #####
4 Thu, 26 Mar 2020 13:32:45 +0000
5
6 INPUT SETTINGS
7
8 Structure:           fcc
9 Size:                2
10 Alat (Bohr):        6.9
11 Alat (Bohr) (exp.): 13.8
12 Alat (Ang):         3.6501000000000006
13 Alat (Ang) (exp.):  7.300200000000001
14 Atoms in crystal:   32
15 Atoms in crystal (exp.): 32
16 AMU per crystal:    863.423999999994
17
18 Ecutfwfc:           50
19 Ecutrho:             200
20 Degauss:             0.04
21 Kpoints:             automatic 11 11 11 1 1 1
22
23 EOS Points:          11
24 Total Strain:         0.005
25
26 V0 (Bohr^3):         111.192295309
27 E0:                  -39.5027342866
28 B0 (RY/BOHR3):       0.00531912635008
29 B0 (GPA):             77.572208693
30 BOP:                 2.0
31
32 EC Points:            9
33 Total Strain:          0.005
34
35
36 Relaxed alat (Bohr):  7.6325799
37 Relaxed alat (Bohr) (exp): 15.2651598
38 Relaxed alat (Ang):     4.0376347671
39 Relaxed alat (Ang) (exp): 8.0752695342
40 Relaxed cp:             1.0   0.0   0.0
41                      0.0   1.0   0.0
42                      0.0   0.0   1.0
43 Relaxed volume (Bohr^3): 3557.16544569
44 Relaxed density (kgm^-3): 2722.71416748
45
46 Stiffness (RY/BOHR3):  0.0076028 0.0039542 0.0039563 0.0   0.0   0.0
47                      0.0039542 0.0075973 0.003945 0.0   0.0   0.0
48                      0.0039563 0.003945 0.0076068 0.0   0.0   0.0
49                      0.0   0.0   0.0   0.0023326 0.0   0.0
50                      0.0   0.0   0.0   0.0   0.0023349 0.0
51                      0.0   0.0   0.0   0.0   0.0   0.0023339
52
53 Stiffness (GPA):      110.877131 57.6666837 57.6978857 0.0   0.0   0.0
54                      57.6666837 110.796070 57.5331112 0.0   0.0   0.0
55                      57.6978857 57.5331112 110.934314 0.0   0.0   0.0
56                      0.0   0.0   0.0   34.0184756 0.0   0.0
57                      0.0   0.0   0.0   0.0   34.0513488 0.0
58                      0.0   0.0   0.0   0.0   0.0   34.037456

```

```

59
60 Compliance (1/GPA):      0.0140133 -0.0048022 -0.0047979 0.0    0.0    0.0
61                      -0.0048022  0.0139977 -0.0047619 0.0    0.0    0.0
62                      -0.0047979 -0.0047619  0.0139794 0.0    0.0    0.0
63                      0.0     -0.0     -0.0     0.0293958 -0.0   -0.0
64                      0.0     0.0     0.0     0.0     0.0293674 0.0
65                      0.0     0.0     0.0     0.0     0.0     0.0293794
66
67 Bulk Modulus B (GPA):    75.378064563
68 BR (GPA):                75.3780315039
69 BV (GPA):                75.378097622
70
71 Shear Modulus G (GPA):   30.8455908955
72 GR:                     30.6224032645
73 GV:                     31.0687785264
74
75 Young's Modulus E (GPA): 81.4294731854
76
77 Poisson's Ratio ν:       0.319953205977
78
79
80 L Elastic Wave V:       0.206857953612
81 T Elastic Wave V:       0.106437709845
82 M Elastic Wave V:       0.119193708075
83
84 Debye Temperature:      0.00139515659606
85
86 Melting Point:          1300.0K

```

N.2.4 Known and Calculated Values

	Al Experimental	Al DFT (this work)
Structure	Face Centered Cubic	Face Centered Cubic
a_0 (Angs)	4.05 Angstrom	4.04 Angstrom
Nearest Neighbour	2.86 Angstrom	2.86 Angstrom
Basis vectors	$\begin{bmatrix} 1.0 & 0.0 & 0.0 \\ 0.0 & 1.0 & 0.0 \\ 0.0 & 0.0 & 1.0 \end{bmatrix}$	$\begin{bmatrix} 1.0 & 0.0 & 0.0 \\ 0.0 & 1.0 & 0.0 \\ 0.0 & 0.0 & 1.0 \end{bmatrix}$
E_{coh} (eV)	-3.36 eV	Not Calculated
Bulk Modulus B_0 (GPA)	76	77.6
Bulk Modulus $B_{0,r}$ (GPA)	-	75.4
Bulk Modulus $B_{0,g}$ (GPA)	-	75.4
Young's modulus E (GPA)	70	81.4
Shear Modulus G (GPA)	26	30.8
Poisson Ratio ν	0.35	0.32
Elastic Constants (GPA)	$\begin{bmatrix} 114 & 62 & 62 & 0 & 0 & 0 \\ 62 & 114 & 62 & 0 & 0 & 0 \\ 62 & 62 & 114 & 0 & 0 & 0 \\ 0 & 0 & 0 & 32 & 0 & 0 \\ 0 & 0 & 0 & 0 & 32 & 0 \\ 0 & 0 & 0 & 0 & 0 & 32 \end{bmatrix}$	$\begin{bmatrix} 110.9 & 57.7 & 57.7 & 0 & 0 & 0 \\ 57.7 & 110.8 & 57.5 & 0 & 0 & 0 \\ 57.7 & 57.5 & 110.9 & 0 & 0 & 0 \\ 0 & 0 & 0 & 34.0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 34.0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 34.0 \end{bmatrix}$

Table N.2: FCC Aluminium: Experimental Values vs DFT Calculated Variables

N.3 BCC Iron [No Magnetism]

N.3.1 Major DFT Settings

Ecutwfc: 71

Ecutrho: 430

Smearing: 0.04

K-points: 9 9 9 1 1 1

Nspin: 1 (non-polarized calculation)

Pseudopotential: Fe.pbe-spn-kjpaw-psl.1.0.0.UPF

N.3.2 Equation of State and Elastic Constants

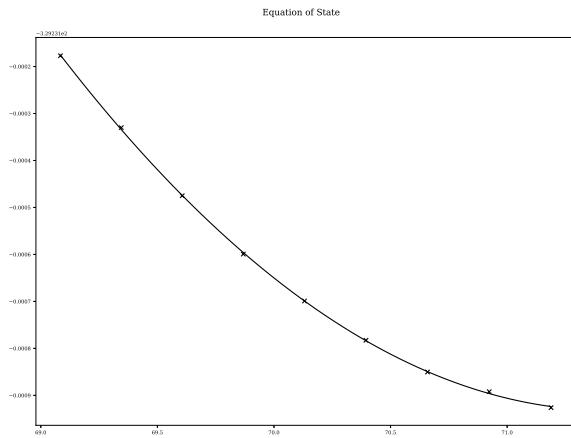


Figure N.3: Iron BCC equation of state plot

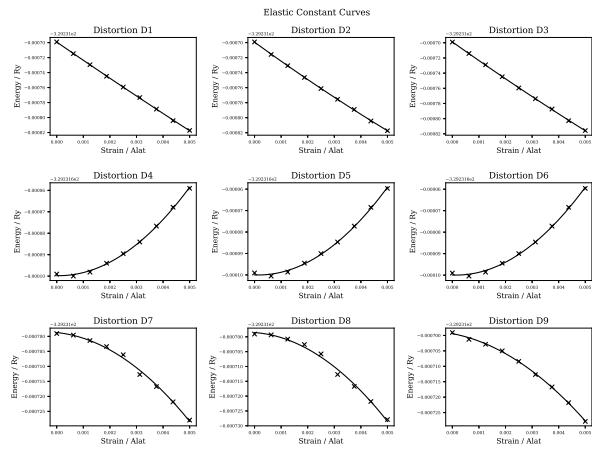


Figure N.4: Iron BCC elastic constants plot

N.3.3 Iron BCC Results File

Listing N.2: Iron BCC DFT Calculated Properties

```

1 ######
2 #          RESULTS          #
3 #####
4 Tue, 03 Nov 2020 06:01:22 +0000
5
6 INPUT SETTINGS
7 #####
8
9 Structure:           bcc
10 Size:                2
11 AO:                 2.8
12 AO Units:           ang
13 AO (Bohr):          10.584
14 AO (Angs):           5.598936
15
16 Ecutwfc:             71
17 Ecutrho:              430
18 Degauss:              0.04
19 Kpoints:              9 9 9 1 1 1
20 Kpoints Type:         automatic
21 EOS Points:           9
22 EOS Strain:            0.005
23 EC Points:             9
24 EC Strain:              0.005
25
26
27 RELAXED
28 #####
29
30 a0 (Bohr):           10.391466456
31 CP:                  1.0    0.0    0.0
32                      0.0    1.0    0.0
33                      0.0    0.0    1.0
34 Volume (Bohr^3/unit cell): 1122.09730705
35 Density KG/m^3:        8932.14485799
36 AMU per crystal unit: 893.5200000000003
37 Atoms per crystal unit: 16
38 Total Energy/Ry:       -5267.70721439
39 Energy per Atom/Ry:    -329.231700899375
40 Total Force Ry/Bohr:   1.7e-05
41 Force per atom Ry/Bohr: 1.0625e-06
42
43
44 a0 primitive (Bohr):   5.195733228
45 a0 primitive (Ang):     2.748542877612
46
47
48 EOS
49 #####
50
51 V0 (Bohr^3 / atom):    71.4368199046
52 E0 (Ry / atom):         -329.231932186
53 B0 (RY/BOHR3):          0.0195284174344
54 B0 (GPA):               287.272784669
55 BOP:                   4.98933790197
56
57
58 EC

```

```

59 #####
60
61 Stiffness (RY/BOHR3):   0.0043229 0.0225755 0.0216226 0.0      0.0      0.0
62                      0.0225755 0.0121066 0.0217004 0.0      0.0      0.0
63                      0.0216226 0.0217004 0.0083439 0.0      0.0      0.0
64                      0.0      0.0      0.0      0.0121324 0.0      0.0
65                      0.0      0.0      0.0      0.0      0.0123943 0.0
66                      0.0      0.0      0.0      0.0      0.0      0.0123441
67
68 Stiffness (GPA):       63.04417 329.233427 315.336007 0.0      0.0      0.0
69                      329.233427 176.558337 316.470988 0.0      0.0      0.0
70                      315.336007 316.470988 121.684505 0.0      0.0      0.0
71                      0.0      0.0      0.0      176.934632 0.0      0.0
72                      0.0      0.0      0.0      0.0      180.753497 0.0
73                      0.0      0.0      0.0      0.0      0.0      180.022531
74
75 Compliance (1/GPA): -0.0026218 0.0019907 0.001617 0.0      0.0      0.0
76                      0.0019907 -0.0030583 0.0027951 0.0      0.0      0.0
77                      0.001617 0.0027951 -0.0032415 0.0      0.0      0.0
78                      0.0      0.0      0.0      0.0056518 0.0      0.0
79                      0.0      0.0      0.0      0.0      0.0055324 0.0
80                      0.0      0.0      0.0      0.0      0.0      0.0055549
81
82 Stability:
83 C11:                  63.044169998 (Stable)
84 C11C22 - C12C12:     -97263.6756949 (Unstable)
85 C11*C22*C33+2*C12*C13*C23-C11*C23*C23-C33*C12*C12: 47561899.9206 (Stable)
86 C44:                  176.934632292 (Stable)
87 C55:                  180.753497554 (Stable)
88 C66:                  180.022531471 (Stable)
89
90 Bulk Modulus B (GPA): 255.592980231
91 BR (GPA):             257.478420517
92 BV (GPA):             253.707539944
93
94 Shear Modulus G (GPA): -643.096162505
95 GR:                   -1353.7508966
96 GV:                   67.5585715904
97
98 Young's Modulus E (GPA): -11960.7418627
99
100 Poisson's Ratio v:    8.29934165372
101
102
103 L Elastic Wave V:    nan
104 T Elastic Wave V:    nan
105 M Elastic Wave V:    nan
106
107 Debye Temperature:   nan
108
109 Melting Point:       1357.0K
110
111
112
113
114
115
116
117
118 References
119 =====

```

- 120
121 First Principles Calculations of Elastic Properties of Metals
122 M. J. Mehl, B. M. Klein, D. A. Papaconstantopoulos
123 1994
124
125 Ab Initio Study of the Elastic and Mechanical Properties of B19 TiAl
126 Y. Wen, L. Wang, H. Liu and L. Song
127 Crystals
128 2017
129
130 Density functional theory for calculation of elastic properties of orthorhombic crystals - applications to TiSi₂
131 P. Ravindran, Lars Fast, P. A. Korzhavyi, B. Johansson
132 Journal of Applied Physics
133 1998
-

N.4 BCC Iron [Ferromagnetic]

N.4.1 Major DFT Settings

Setting	Value
ecutwfc (Ry)	71
ecutrho	430
smearing (Ry)	0.04
k-points	9 9 9 1 1 1
nspin	0 (Non-magnetic) and 2 (Collinear Spin)
pseudopotential	Fe.pbe-spn-kjpaw_psl.1.0.0.UPF
etot_conv_thr	0.0001
forc_conv_thr	0.001
conv_thr	1.0D-6
diagonalization	david
mixing_beta	0.1
mixing_mode	plain

Table N.3: Iron DFT settings

N.4.2 Equation of State and Elastic Constants

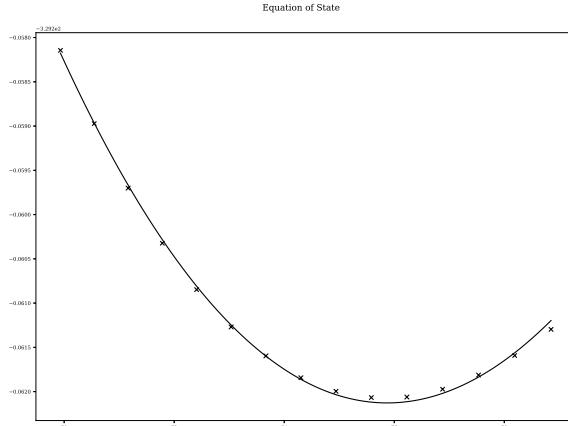


Figure N.5: Iron BCC (magnetic) equation of state plot

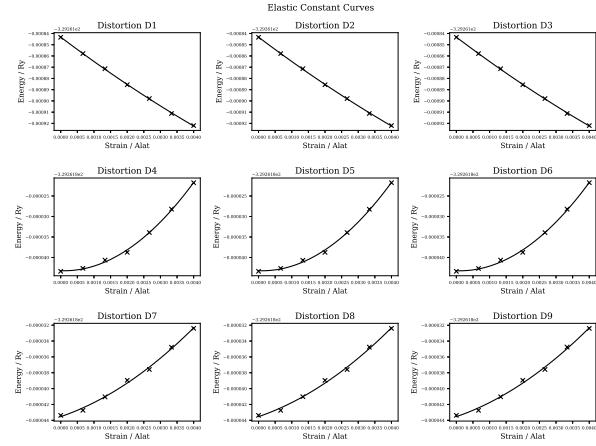


Figure N.6: Iron BCC (magnetic) elastic constants plot

N.4.3 Iron Ferromagnetic BCC Results File

Listing N.3: Iron BCC DFT calculated properties

```

1 ##########
2 #          RESULTS          #
3 ##########
4
5 INPUT SETTINGS
6 #####
7
8 Structure:           bcc
9 Size:                2
10 AO:                 2.8
11 AO Units:           ang
12 AO (Bohr):          10.584
13 AO (Angs):           5.598936
14
15 Ecutfc:              71
16 Ecutrho:             430
17 Degauss:              0.04
18 Kpoints:              9 9 9 1 1 1
19 Kpoints Type:         automatic
20 EOS Points:           15
21 EOS Strain:            0.02
22 EC Points:             7
23 EC Strain:             0.004
24
25
26 RELAXED
27 #####
28
29 a0 (Bohr):           10.5935195777
30 CP:                  1.0   0.0   0.0
31                      0.0   1.0   0.0
32                      0.0   0.0   1.0
33 Volume (Bohr^3/unit cell): 1188.83291445
34 Density KG/m^3:          8430.73536197
35 AMU per crystal unit:    893.5200000000003
36 Atoms per crystal unit:  16
37 Total Energy/Ry:          -5268.18846365
38 Energy per Atom/Ry:        -329.261778978125
39 Total Force Ry/Bohr:      2.2e-05
40 Force per atom Ry/Bohr:   1.375e-06
41
42
43 a0 primitive (Bohr):     5.2967597888519995
44 a0 primitive (Ang):        2.8019859283027078
45
46
47 EOS
48 #####
49
50 V0 (Bohr^3 / atom):      75.8702665574
51 E0 (Ry / atom):           -329.262127846
52 B0 (RY/BOHR3):            0.0162627705061
53 B0 (GPA):                 239.23348553
54 BOP:                      3.31889649178
55
56
57 EC
58 #####

```

59
60 Stiffness (RY/BOHR3): 0.0171516 0.0124976 0.0125676 0.0 0.0 0.0
61 0.0124976 0.0170822 0.0125444 0.0 0.0 0.0
62 0.0125676 0.0125444 0.0172259 0.0 0.0 0.0
63 0.0 0.0 0.0 0.0095442 0.0 0.0
64 0.0 0.0 0.0 0.0 0.0095793 0.0
65 0.0 0.0 0.0 0.0 0.0 0.0095537
66
67 Stiffness (GPA): 250.132354 182.259753 183.281653 0.0 0.0 0.0
68 182.259753 249.120311 182.943207 0.0 0.0 0.0
69 183.281653 182.943207 251.216687 0.0 0.0 0.0
70 0.0 0.0 0.0 139.188774 0.0 0.0
71 0.0 0.0 0.0 0.0 139.701368 0.0
72 0.0 0.0 0.0 0.0 0.0 139.328420
73
74 Compliance (1/GPA): 0.0104302 -0.0043908 -0.0044121 0.0 0.0 0.0
75 -0.0043908 0.0104768 -0.0044261 0.0 0.0 0.0
76 -0.0044121 -0.0044261 0.0104228 0.0 0.0 0.0
77 0.0 0.0 0.0 0.0071845 0.0 0.0
78 0.0 0.0 0.0 0.0 0.0071581 0.0
79 0.0 0.0 0.0 0.0 0.0 0.0071773
80
81 Stability:
82 C11: 250.132354807 (Stable)
83 C11C22 - C12C12: 29094.4322943 (Stable)
84 C11*C22*C33+2*C12*C13*C23-C11*C23*C23-C33*C12*C12: 11159910.6986 (Stable)
85 C44: 139.188774163 (Stable)
86 C55: 139.701368243 (Stable)
87 C66: 139.328420919 (Stable)
88
89 Bulk Modulus B (GPA): 205.266912094
90 BR (GPA): 205.262870468
91 BV (GPA): 205.270953721
92
93 Shear Modulus G (GPA): 79.4449473647
94 GR: 61.7805328168
95 GV: 97.1093619125
96
97 Young's Modulus E (GPA): 211.100586016
98
99 Poisson's Ratio v: 0.328596676183
100
101
102 L Elastic Wave V: 0.192124405208
103 T Elastic Wave V: 0.0970734382597
104 M Elastic Wave V: 0.108830261337
105
106 Debye Temperature: 0.000728462251696
107
108 Melting Point: 2076.0K
109
110
111
112
113
114
115
116
117 References
118 =====
119

- 120 First Principles Calculations of Elastic Properties of Metals
121 M. J. Mehl, B. M. Klein, D. A. Papaconstantopoulos
122 1994
123
124 Ab Initio Study of the Elastic and Mechanical Properties of B19 TiAl
125 Y. Wen, L. Wang, H. Liu and L. Song
126 Crystals
127 2017
128
129 Density functional theory for calculation of elastic properties of orthorhombic crystals - applications to TiSi₂
130 P. Ravindran, Lars Fast, P. A. Korzhavyi, B. Johansson
131 Journal of Applied Physics
132 1998
-

N.4.4 Known and Calculated Results

	Fe Experimental	Fe DFT No Magnetism (this work)	Fe DFT Ferromagnetic (this work)
Structure	Body Centered Cubic	Body Centered Cubic	
a_0 (Angs)	2.86 Angstrom[143]	2.75 Angstrom	2.80 Angstrom
Nearest Neighbour	2.48 Angstrom[143]	2.38 Angstrom	2.42 Angstrom
Basis vectors	$\begin{bmatrix} 1.0 & 0.0 & 0.0 \\ 0.0 & 1.0 & 0.0 \\ 0.0 & 0.0 & 1.0 \end{bmatrix}$	$\begin{bmatrix} 1.0 & 0.0 & 0.0 \\ 0.0 & 1.0 & 0.0 \\ 0.0 & 0.0 & 1.0 \end{bmatrix}$	$\begin{bmatrix} 1.0 & 0.0 & 0.0 \\ 0.0 & 1.0 & 0.0 \\ 0.0 & 0.0 & 1.0 \end{bmatrix}$
E_{coh} (eV)	-4.32 [143]	Not Calculated	Not Calculated
Bulk Modulus B_0 (GPA)	170	287.3	239.2
Bulk Modulus $B_{0,r}$ (GPA)	-	257.5	205.3
Bulk Modulus $B_{0,g}$ (GPA)	-	253.7	205.3
Young's modulus E (GPA)	211	-11960.7	211.1
Shear Modulus G (GPA)	82	-643.1	79.4
Poisson Ratio ν	0.29	8.29	0.33
Elastic Constants (GPA)	$\begin{bmatrix} 243 & 145 & 145 & 0 & 0 & 0 \\ 145 & 243 & 145 & 0 & 0 & 0 \\ 145 & 145 & 243 & 0 & 0 & 0 \\ 0 & 0 & 0 & 116 & 0 & 0 \\ 0 & 0 & 0 & 0 & 116 & 0 \\ 0 & 0 & 0 & 0 & 0 & 116 \end{bmatrix}$ [143]	$\begin{bmatrix} 63.0 & 329.2 & 315.3 & 0 & 0 & 0 \\ 329.2 & 176.6 & 316.4 & 0 & 0 & 0 \\ 315.3 & 316.5 & 121.7 & 0 & 0 & 0 \\ 0 & 0 & 0 & 176.9 & 0 & 0 \\ 0 & 0 & 0 & 0 & 180.8 & 0 \\ 0 & 0 & 0 & 0 & 0 & 180.0 \end{bmatrix}$	$\begin{bmatrix} 250.1 & 182.3 & 183.3 & 0 & 0 & 0 \\ 182.3 & 249.1 & 182.9 & 0 & 0 & 0 \\ 183.2 & 182.9 & 251.2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 139.2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 139.7 & 0 \\ 0 & 0 & 0 & 0 & 0 & 139.3 \end{bmatrix}$

Table N.4: BCC Iron: Experimental Values vs DFT Calculated Variables

N.5 FCC Iron [Antiferromagnetic]

N.5.1 Major DFT Settings

Setting	Value
ecutwfc (Ry)	71
ecutrho	430
smearing (Ry)	0.04
k-points	9 9 9 1 1 1
nspin	2 (Collinear Spin)
pseudopotential	Fe.pbe-spn-kjpaw_psl.1.0.0.UPF
etot_conv_thr	0.0001
forc_conv_thr	0.001
conv_thr	1.0D-6
diagonalization	david
mixing_beta	0.1
mixing_mode	plain

Table N.5: Iron DFT settings

N.5.2 Equation of State and Elastic Constants

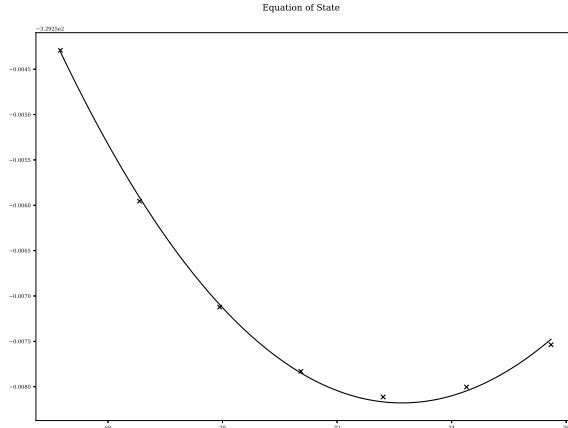


Figure N.7: Iron FCC (magnetic) equation of state plot

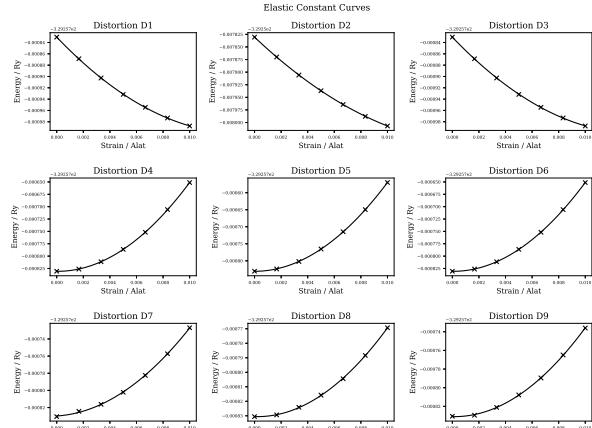


Figure N.8: Iron FCC (magnetic) elastic constants plot

N.5.3 Iron FCC Results File

Listing N.4: Iron FCC DFT Calculated Properties

```

1 ######
2 #          RESULTS          #
3 ######
4
5 INPUT SETTINGS
6 #####
7
8 Structure:           fcc
9 Size:                2
10 AO:                 3.4278156
11 AO Units:           ang
12 AO (Bohr):          12.957142968
13 AO (Angs):           6.854328630072
14
15 Ecutwfc:             71
16 Ecutrho:              430
17 Degauss:              0.04
18 Kpoints:              9 9 9 1 1 1
19 Kpoints Type:         automatic
20 EOS Points:           7
21 EOS Strain:            0.02
22 EC Points:              7
23 EC Strain:              0.01
24
25
26 RELAXED
27 #####
28
29 a0 (Bohr):            12.9381866998
30 CP:                  1.0    0.0    0.0
31                      0.0    1.05447369502   0.0
32                      0.0    0.0    0.999986980953
33 Volume (Bohr^3/unit cell): 2283.75934496
34 Density KG/m^3:          8777.4009231
35 AMU per crystal unit:   1787.040000000006
36 Atoms per crystal unit: 32
37 Total Energy/Ry:         -10536.25057837
38 Energy per Atom/Ry:      -329.2578305740625
39 Total Force Ry/Bohr:     5.1e-05
40 Force per atom Ry/Bohr:  1.59375e-06
41
42
43 a0 primitive (Bohr):     6.4690933498955
44 a0 primitive (Ang):       3.4221503820947197
45
46
47 EOS
48 #####
49
50 V0 (Bohr^3 / atom):     73.1321918692
51 E0 (Ry / atom):          -329.258178984
52 B0 (RY/BOHR3):           0.0153732026949
53 B0 (GPA):                226.147498244
54 BOP:                     3.74220549488
55
56
57 EC
58 #####

```

59
60 Stiffness (RY/BOHR3): 0.0250005 0.009711 0.0160292 0.0 0.0 0.0
61 0.009711 0.0204815 0.0089411 0.0 0.0 0.0
62 0.0160292 0.0089411 0.0250019 0.0 0.0 0.0
63 0.0 0.0 0.0 0.0127715 0.0 0.0
64 0.0 0.0 0.0 0.0 0.0182966 0.0
65 0.0 0.0 0.0 0.0 0.0 0.0127778
66
67 Stiffness (GPA): 364.598330 141.621966 233.764448 0.0 0.0 0.0
68 141.621966 298.695202 130.394083 0.0 0.0 0.0
69 233.764448 130.394083 364.619131 0.0 0.0 0.0
70 0.0 0.0 0.0 186.254349 0.0 0.0
71 0.0 0.0 0.0 0.0 266.830684 0.0
72 0.0 0.0 0.0 0.0 0.0 186.346168
73
74 Compliance (1/GPA): 0.004966 -0.0011431 -0.002775 0.0 0.0 0.0
75 -0.0011431 0.0042304 -0.00078 0.0 0.0 0.0
76 -0.002775 -0.00078 0.0048006 0.0 0.0 0.0
77 0.0 0.0 0.0 0.005369 0.0 0.0
78 0.0 0.0 0.0 0.0 0.0037477 0.0
79 0.0 0.0 0.0 0.0 0.0 0.0053664
80
81 Stability:
82 C11: 364.598330439 (Stable)
83 C11C22 - C12C12: 88846.9908456 (Stable)
84 C11*C22*C33+2*C12*C13*C23-C11*C23*C23-C33*C12*C12: 34829887.0518 (Stable)
85 C44: 186.254349116 (Stable)
86 C55: 266.830684551 (Stable)
87 C66: 186.346168623 (Stable)
88
89 Bulk Modulus B (GPA): 221.980853006
90 BR (GPA): 217.35352142
91 BV (GPA): 226.608184591
92
93 Shear Modulus G (GPA): 144.783544731
94 GR: 126.872037967
95 GV: 162.695051494
96
97 Young's Modulus E (GPA): 356.782113599
98
99 Poisson's Ratio v: 0.232122456533
100
101
102 L Elastic Wave V: 0.217447522685
103 T Elastic Wave V: 0.128433002219
104 M Elastic Wave V: 0.142291445915
105
106 Debye Temperature: 0.0009653163572
107
108 Melting Point: 2528.0K
109
110
111
112
113
114
115
116
117 References
118 =====
119

- 120 First Principles Calculations of Elastic Properties of Metals
 121 M. J. Mehl, B. M. Klein, D. A. Papaconstantopoulos
 122 1994
 123
 124 Ab Initio Study of the Elastic and Mechanical Properties of B19 TiAl
 125 Y. Wen, L. Wang, H. Liu and L. Song
 126 Crystals
 127 2017
 128
 129 Density functional theory for calculation of elastic properties of orthorhombic crystals - applications to TiSi₂
 130 P. Ravindran, Lars Fast, P. A. Korzhavyi, B. Johansson
 131 Journal of Applied Physics
 132 1998
-

Iron FCC (this work)		
Structure	Face Centered Cubic	Face Centered Cubic
a_0 (Angs)	-	3.42 Angstrom
Nearest Neighbour	-	2.86 Angstrom
Basis vectors	$\begin{bmatrix} 1.0 & 0.0 & 0.0 \\ 0.0 & 1.0 & 0.0 \\ 0.0 & 0.0 & 1.0 \end{bmatrix}$	$\begin{bmatrix} 1.0 & 0.0 & 0.0 \\ 0.0 & 1.0 & 0.0 \\ 0.0 & 0.0 & 1.0 \end{bmatrix}$
E_{coh} (eV)	-	-4.26
Bulk Modulus B_0 (GPA)	-	226.1
Bulk Modulus $B_{0,r}$ (GPA)	-	217.3
Bulk Modulus $B_{0,v}$ (GPA)	-	226.6
Young's modulus E (GPA)	-	356.8
Shear Modulus G (GPA)	-	144.8
Poisson Ratio ν	-	0.23
Elastic Constants (GPA)	-	$\begin{bmatrix} 364.6 & 141.6 & 233.8 & 0 & 0 & 0 \\ 141.6 & 298.7 & 130.4 & 0 & 0 & 0 \\ 233.8 & 130.4 & 364.6 & 0 & 0 & 0 \\ 0 & 0 & 0 & 186.3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 266.8 & 0 \\ 0 & 0 & 0 & 0 & 0 & 186.3 \end{bmatrix}$

Table N.6: Iron FCC: DFT Calculated Properties

N.6 FCC Palladium

N.6.1 Major DFT Settings

Setting	Value
Ecutwfc (Ry)	71
Ecutrho	430
Smearing (Ry)	0.04
K-points	9 9 9 1 1 1
Nspin	0 (Non-magnetic)
Pseudopotential	Pd.pbe-spn-kjpaw_psl.1.0.0.UPF
etot_conv_thr	0.0001
forc_conv_thr	0.001
conv_thr	1.0D-6
diagonalization	david
mixing_beta	0.1
mixing_mode	plain

Table N.7: Palladium DFT settings

N.6.2 Equation of State and Elastic Constants

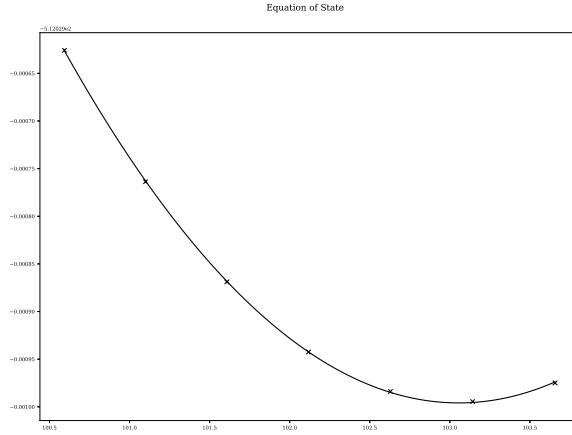


Figure N.9: Palladium FCC equation of state plot

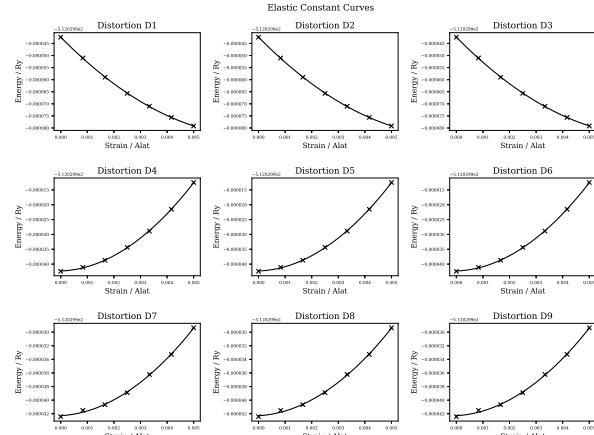


Figure N.10: Palladium FCC elastic constants plot

N.6.3 Palladium FCC Results File

Listing N.5: Palladium FCC DFT Calculated Properties

```

1 ##########
2 #          RESULTS          #
3 ##########
4
5 INPUT SETTINGS
6 #####
7
8 Structure:           fcc
9 Size:                2
10 AO:                 3.89
11 AO Units:          ang
12 AO (Bohr):         14.7042
13 AO (Angs):          7.7785218
14
15 Ecutfc:             71
16 Ecutrho:            430
17 Degauss:            0.04
18 Kpoints:             9 9 9 1 1 1
19 Kpoints Type:       automatic
20 EOS Points:         7
21 EOS Strain:         0.005
22 EC Points:          7
23 EC Strain:          0.005
24
25
26 RELAXED
27 #####
28
29 a0 (Bohr):          14.8394198232
30 CP:                  1.0   0.0   0.0
31                      0.0   1.0   0.0
32                      0.0   0.0   1.0
33 Volume (Bohr^3/unit cell): 3267.76460963
34 Density KG/m^3:        11689.7296625
35 AMU per crystal unit: 3405.4400000000014
36 Atoms per crystal unit: 32
37 Total Energy/Ry:      -16384.95810846
38 Energy per Atom/Ry:    -512.029940889375
39 Total Force Ry/Bohr:   0.000188
40 Force per atom Ry/Bohr: 5.875e-06
41
42
43 a0 primitive (Bohr):   7.4197099116
44 a0 primitive (Ang):     3.9250265432364
45
46
47 EOS
48 #####
49
50 V0 (Bohr^3 / atom):   103.055133214
51 E0 (Ry / atom):        -512.029995887
52 B0 (RY/BOHR3):         0.012538146479
53 B0 (GPA):              184.442403779
54 BOP:                  4.7014295893
55
56
57 EC
58 #####

```

59
60 Stiffness (RY/BOHR3): 0.0149806 0.010383 0.0103812 0.0 0.0 0.0
61 0.010383 0.0149801 0.0103722 0.0 0.0 0.0
62 0.0103812 0.0103722 0.0149778 0.0 0.0 0.0
63 0.0 0.0 0.0 0.0055032 0.0 0.0
64 0.0 0.0 0.0 0.0 0.0055036 0.0
65 0.0 0.0 0.0 0.0 0.0 0.0055024
66
67 Stiffness (GPA): 218.471929 151.422406 151.395627 0.0 0.0 0.0
68 151.422406 218.464281 151.264808 0.0 0.0 0.0
69 151.395627 151.264808 218.430622 0.0 0.0 0.0
70 0.0 0.0 0.0 80.2564574 0.0 0.0
71 0.0 0.0 0.0 0.0 80.2629605 0.0
72 0.0 0.0 0.0 0.0 0.0 80.2449828
73
74 Compliance (1/GPA): 0.0105867 -0.0043366 -0.0043346 0.0 0.0 0.0
75 -0.0043366 0.0105705 -0.0043144 0.0 0.0 0.0
76 -0.0043346 -0.0043144 0.0105702 0.0 0.0 0.0
77 0.0 0.0 0.0 0.0124601 0.0 0.0
78 0.0 0.0 0.0 0.0 0.012459 0.0
79 0.0 0.0 0.0 0.0 0.0 0.0124618
80
81 Stability:
82 C11: 218.471929458 (Stable)
83 C11C22 - C12C12: 24799.5680651 (Stable)
84 C11*C22*C33+2*C12*C13*C23-C11*C23*C23-C33*C12*C12: 7353517.38556 (Stable)
85 C44: 80.256457365 (Stable)
86 C55: 80.262960524 (Stable)
87 C66: 80.244982752 (Stable)
88
89 Bulk Modulus B (GPA): 173.725818131
90 BR (GPA): 173.725800894
91 BV (GPA): 173.725835369
92
93 Shear Modulus G (GPA): 56.5595512391
94 GR: 51.5472895795
95 GV: 61.5718128987
96
97 Young's Modulus E (GPA): 153.067378318
98
99 Poisson's Ratio v: 0.35315234089
100
101
102 L Elastic Wave V: 0.145988361768
103 T Elastic Wave V: 0.0695585857774
104 M Elastic Wave V: 0.0782389075958
105
106 Debye Temperature: 0.000471028127273
107
108 Melting Point: 1910.0K
109
110
111
112
113
114
115
116
117 References
118 =====
119

120 First Principles Calculations of Elastic Properties of Metals
 121 M. J. Mehl, B. M. Klein, D. A. Papaconstantopoulos
 122 1994
 123
 124 Ab Initio Study of the Elastic and Mechanical Properties of B19 TiAl
 125 Y. Wen, L. Wang, H. Liu and L. Song
 126 Crystals
 127 2017
 128
 129 Density functional theory for calculation of elastic properties of orthorhombic crystals - applications to TiSi₂
 130 P. Ravindran, Lars Fast, P. A. Korzhavyi, B. Johansson
 131 Journal of Applied Physics
 132 1998

N.6.4 Known and Calculated Values

	Palladium FCC	Palladium FCC (this work)
Structure	Face Centered Cubic	Face Centered Cubic
a_0 (Angs)	3.89	3.92
Nearest Neighbour	2.75	2.77
Basis vectors	$\begin{bmatrix} 1.0 & 0.0 & 0.0 \\ 0.0 & 1.0 & 0.0 \\ 0.0 & 0.0 & 1.0 \end{bmatrix}$	$\begin{bmatrix} 1.0 & 0.0 & 0.0 \\ 0.0 & 1.0 & 0.0 \\ 0.0 & 0.0 & 1.0 \end{bmatrix}$
E_{coh} (eV)	-3.91	not calculated
Bulk Modulus B_0 (GPA)	195.5	184.4
Bulk Modulus $B_{0,r}$ (GPA)	-	173.7
Bulk Modulus $B_{0,v}$ (GPA)	-	173.7
Young's modulus E (GPA)	121	153.1
Shear Modulus G (GPA)	44	56.6
Poisson Ratio ν	0.39	0.35
Elastic Constants (GPA)	$\begin{bmatrix} 234 & 176.1 & 176.1 & 0 & 0 & 0 \\ 176.1 & 234 & 176.1 & 0 & 0 & 0 \\ 176.1 & 176.1 & 234 & 0 & 0 & 0 \\ 0 & 0 & 0 & 71.2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 71.2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 71.2 \end{bmatrix}$	$\begin{bmatrix} 218.5 & 151.4 & 151.4 & 0 & 0 & 0 \\ 151.4 & 218.5 & 151.3 & 0 & 0 & 0 \\ 151.4 & 151.3 & 218.5 & 0 & 0 & 0 \\ 0 & 0 & 0 & 80.3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 80.3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 80.3 \end{bmatrix}$

Table N.8: FCC Palladium: Experimental Values vs DFT Calculated Properties

N.7 FCC Ruthenium

N.7.1 Major DFT Settings

Ecutwfc: 71
 Ecutrho: 430
 Smearing: 0.04
 K-points: 9 9 9 1 1 1
 Nspin: 1 (non-polarized calculation)
 Pseudopotential: Ru 101.07 Ru.pbe-spn-kjpaw_psl.1.0.0.UPF

N.7.2 Equation of State and Elastic Constants

Setting	Value
ecutwfc (Ry)	71
ecutrho	430
smearing (Ry)	0.04
k-points	9 9 9 1 1 1
nspin	2 (Collinear Spin)
pseudopotential	Ru.pbe-spn-kjpaw_psl.1.0.0.UPF
etot_conv_thr	0.0001
forc_conv_thr	0.001
conv_thr	1.0D-6
diagonalization	david
mixing_beta	0.1
mixing_mode	plain

Table N.9: Ruthenium DFT settings

N.7.3 Ruthenium FCC Results File

Listing N.6: Ruthenium FCC DFT Calculated Properties

```

1 ######
2 #          RESULTS          #
3 #####
4 Fri, 26 Nov 2021 04:44:37 +0000
5
6 INPUT SETTINGS
7 #####
8
9 Structure Type:      fcc
10 Size:                2
11 AO:                  4.0
12 AO Units:            angs
13 AO (Bohr):           15.12
14 AO (Angs):            7.99848
15
16 Ecutwfc:              71
17 Ecutrho:               430
18 Degauss:               0.04
19 Kpoints:               9 9 9 1 1 1
20 Kpoints Type:         automatic
21 EOS Points:            11
22 EOS Strain:             0.005
23 EC Points:              9
24 EC Strain:               0.005
25
26
27 RELAXED
28 #####
29
30 a0 (Bohr):            14.4007416
31 CP:                   1.0   0.0   0.0
32                      0.0   1.0   0.0
33                      0.0   0.0   1.0
34 Volume (Bohr^3/unit cell): 2986.445358287136
35 Density KG/m^3:          12147.856996566989
36 AMU per crystal unit:    3234.240000000001
37 Atoms per crystal unit:   32
38 Total Energy/Ry:          -13813.30088106
39 Energy per Atom/Ry:        -431.665652533125
40 Total Force Ry/Bohr:       0.000137
41 Force per atom Ry/Bohr:    4.28125e-06
42
43
44 a0 primitive (Bohr):      7.2003708
45 a0 primitive (Ang):        3.8089961532000003
46
47
48 EOS
49 #####
50
51 V0 (Bohr^3 / atom):       93.41718637560044
52 E0 (Ry / atom):            -431.66565367812325
53 B0 (RY/BOHR3):             0.020926676273491183
54 B0 (GPA):                  307.84187132119206
55 BOP:                      2.2239400402684173
56
57
58 EC

```

59 #####
60
61 Stiffness (RY/BOHR3): 0.0323332 0.0150236 0.0150295 0.0 0.0 0.0
62 0.0150236 0.032334 0.0150304 0.0 0.0 0.0
63 0.0150295 0.0150304 0.0323458 0.0 0.0 0.0
64 0.0 0.0 0.0 0.0167987 0.0 0.0
65 0.0 0.0 0.0 0.0 0.0167978 0.0
66 0.0 0.0 0.0 0.0 0.0 0.0167975
67
68 Stiffness (GPA): 471.535040 219.099143 219.185184 0.0 0.0 0.0
69 219.099143 471.546672 219.197356 0.0 0.0 0.0
70 219.185184 219.197356 471.719577 0.0 0.0 0.0
71 0.0 0.0 0.0 244.986869 0.0 0.0
72 0.0 0.0 0.0 0.0 244.973541 0.0
73 0.0 0.0 0.0 0.0 0.0 244.969279
74
75 Compliance (1/GPA): 0.0030075 -0.0009538 -0.0009542 0.0 0.0 0.0
76 -0.0009538 0.0030075 -0.0009543 0.0 0.0 0.0
77 -0.0009542 -0.0009543 0.0030067 0.0 0.0 0.0
78 0.0 0.0 0.0 0.0040819 0.0 0.0
79 0.0 0.0 0.0 0.0 0.0040821 0.0
80 0.0 0.0 0.0 0.0 0.0 0.0040821
81
82 Stability:
83 C11: 471.535040254 (Stable)
84 C11C22 - C12C12: 174346.3443625905 (Stable)
85 C11*C22*C33+2*C12*C13*C23-C11*C23*C23-C33*C12*C12: 80639667.71993023 (Stable)
86 C44: 244.986869322 (Stable)
87 C55: 244.97354179 (Stable)
88 C66: 244.969279805 (Stable)
89
90 Bulk Modulus B (GPA): 303.30717357998026
91 BR (GPA): 303.3071627547384
92 BV (GPA): 303.3071844052222
93
94 Shear Modulus G (GPA): 187.73207685789976
95 GR: 177.99024187226613
96 GV: 197.47391184353336
97
98 Young's Modulus E (GPA): 466.87258322673495
99
100 Poisson's Ratio v: 0.24345447789438046
101
102
103 L Elastic Wave V: 0.21347878392203118
104 T Elastic Wave V: 0.12431381857023048
105 M Elastic Wave V: 0.1379068118794884
106
107 Debye Temperature: 0.0008555430077179703
108
109 Melting Point: 3072.0K
110
111
112
113
114
115
116
117
118 References
119 =====

120
 121 First Principles Calculations of Elastic Properties of Metals
 122 M. J. Mehl, B. M. Klein, D. A. Papaconstantopoulos
 123 1994
 124
 125 Ab Initio Study of the Elastic and Mechanical Properties of B19 TiAl
 126 Y. Wen, L. Wang, H. Liu and L. Song
 127 Crystals
 128 2017
 129
 130 Density functional theory for calculation of elastic properties of orthorhombic crystals - applications to TiSi₂
 131 P. Ravindran, Lars Fast, P. A. Korzhavyi, B. Johansson
 132 Journal of Applied Physics
 133 1998

N.7.4 Calculated Results

	Ruthenium FCC (this work)						
Structure	Face Centered Cubic						Face Centered Cubic
a_0 (Angs)	-						3.81 Angstrom
Nearest Neighbour	-						2.69 Angstrom
Basis vectors	$\begin{bmatrix} 1.0 & 0.0 & 0.0 \\ 0.0 & 1.0 & 0.0 \\ 0.0 & 0.0 & 1.0 \end{bmatrix}$						$\begin{bmatrix} 1.0 & 0.0 & 0.0 \\ 0.0 & 1.0 & 0.0 \\ 0.0 & 0.0 & 1.0 \end{bmatrix}$
E_{coh} (eV)	-						-6.62
Bulk Modulus B_0 (GPA)	-						307.8
Bulk Modulus $B_{0,r}$ (GPA)	-						303.3
Bulk Modulus $B_{0,v}$ (GPA)	-						303.3
Young's modulus E (GPA)	-						466.9
Shear Modulus G (GPA)	-						187.7
Poisson Ratio ν	-						0.24
Elastic Constants (GPA)	$\begin{bmatrix} 471.54 & 219.10 & 219.20 & 0 & 0 & 0 \\ 219.10 & 471.55 & 219.20 & 0 & 0 & 0 \\ 219.19 & 219.20 & 471.72 & 0 & 0 & 0 \\ 0 & 0 & 0 & 244.99 & 0 & 0 \\ 0 & 0 & 0 & 0 & 244.97 & 0 \\ 0 & 0 & 0 & 0 & 0 & 244.97 \end{bmatrix}$						

Table N.10: Ruthenium FCC: DFT Calculated Properties

Appendix O

DFT Errors, Convergence Failures and Repeat Calculations

O.0.1 Insufficient Memory

The DFT calculations are performed on a supercomputer with many processors and many cores per processor. The program uses two implementations of parallelization.

- OpenMP - many threads have access to the same shared memory
- OpenMPI/MPICH - many processes, each with it's own memory

Whilst some favourable results have been shown using a hybrid of OpenMP and OpenMPI, it was more straightforward to use in OpenMPI mode only in this work. Unfortunately, the amount of memory per process overwhelmed the computing nodes for certain calculations as each process requires a copy of the data held in memory. The job script was modified to reserve entire nodes and all the memory on each node, but to only use the cores on one of the two processors to halve the processes per node, but double the memory per process. It was quicker to do this than wait in the high memory queue on BlueBEAR, where the memory per process would be perhaps double than that required.

O.0.2 Scratch Restrictions

When the calculations first started, the scratch space was located on a large shared space on the BlueBEAR computer. Over time the drive space quota was restricted and the scratch was relocated to individual nodes. There were few issues with less complex calculations as the SCF temporary data was several GB in size with the scratch space on each node being approximately 120GB. Once a batch of collinear iron calculations had been submitted to the job queue, they would quickly fill the scratch drive and the job would terminate after just a few calculations, as the node had run out of scratch drive space.

Listing O.1: Sbatch file

```
1 #!/bin/bash
2 #
3 #SBATCH --job-name=feforfit
4 #SBATCH --output=jobout.txt
5 #SBATCH --account=readmsd02
6 #
7 #SBATCH --ntasks 40
8 #SBATCH --nodes 1
```

```
9 #SBATCH --time 1800:00
10 #SBATCH --mem 180GB
11 #
12 #SBATCH --get-user-env
13 #SBATCH --export=NONE
14 #
15 unset SLURM_EXPORT_ENV
16
17 module purge; module load bluebear
18 module load bear-apps/2018a
19 module load iomkl/2018a
20 module load Python/3.6.3-iomkl-2018a
21 module load matplotlib/2.1.1-iomkl-2018a-Python-3.6.3
22
23 # Change to $PBS_O_WORKDIR
24 cd "$PBS_O_WORKDIR"
25 # Set the number of threads to 1
26 export OMP_NUM_THREADS=1
27 export PROC_COUNT=40
28 export PWSCF_SCRATCH=/scratch/bxp912
29 export PWSCF_PP=/rds/homes/b/bxp912/pp
30 export PWSCF_CACHE=/rds/homes/b/bxp912/pwscf_cache
31 export PWSCF_BIN=/rds/homes/b/bxp912/apps/qe-6.3/bin/pw.x
32 export PWSCF_SCRIPT=/rds/homes/b/bxp912/apps/qe-6.3/bin/pw.sh
33
34 python /rds/homes/b/bxp912/apps/python/qeforfit.py input.in > results.txt
```

Listing O.2: Wrapper file

```
1#!/bin/bash
2rm -R $PWSCF_SCRATCH/*
3mpirun -n $1 $PWSCF_BIN -in $2 > $3
```

As the convergence and equation of state/elastic constant python programs automatically created and submitted the PWscf jobs, they would need to be continually restarted after clearing out the scratch. To work around this a batch script (listing O.1) and wrapper script (listing O.2) for pw.sh were written. The batch script sets the environment variables required for the sbatch job scheduling program and the wrapper script, and the wrapper script then clears the scratch directory before running PWscf.

O.0.3 Caching Repeat Calculations

Several of the programs required many hundreds of DFT calculations. At first, if an error was encountered and the program had to start from scratch, computation time would be wasted. To solve this, a directory was set to cache input and output files for PWscf. A hash would be generated from the input file and this would be used as the cache file name. If a calculation was successful, it would save the input and output file. In future, if the same calculation was submitted again, the cached file would be returned. The cache could be copied between BlueBEAR and other computers to save on repeat submission to the BlueBEAR job scheduler.

O.0.4 Adjustment of DFT Parameters

Throughout the DFT segment of this work, there were a number of issues that either resulted in the calculations failing or completing and not converging. There were a number of causes for this and several solutions were used throughout the process to converge the SCF calculation.

Where the ecutwfc and ecutrho parameters are not converged there is a danger that values too low therefore containing too few plane waves in the basis set. This might cause problems with the SCF causing it not to

converge. The same can be said about a failure to pick appropriate k-points and smearing.

There are a selection of mixing modes available in PWscf, giving a choice of Thomas-Fermi screening for homogeneous systems, local density dependent Thomas Fermi (local-TF) screening for inhomogeneous screening and Broyden mixing as the default.

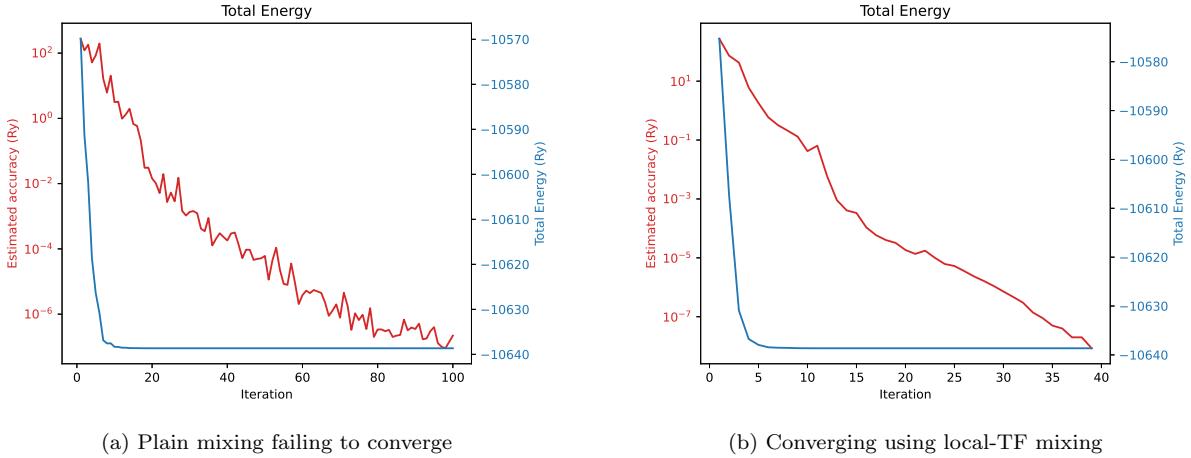


Figure O.1: Fe-Ru calculations failed to converge in 100 iterations when plain mixing was used, but achieved convergence in under 40 iterations with local-TF convergence

For calculations with defects the local-TF setting was used and for equation of state and elastic constant calculations, where the distribution of atoms is more homogeneous, plain Broyden mixing was used. In most circumstances the mixing factor (mixing_beta) was reduced to 0.1. This wasn't always the case, and for certain defect calculations the default value of 0.3 proved to be the more efficient choice.

One other important parameter, that primarily affects geometry and position relaxation calculations (VC-RELAX), was the convergence threshold for the SCF calculations (conv_thr). The default value is 1.0×10^{-6} Ry and this was sufficient for SCF calculations and most VC-RELAX calculations. Several relaxation calculations failed to converge and, at the suggestion of authors of Quantum Espresso, the threshold was reduced to 1.0×10^{-7} Ry. This would lead to a longer time to compute the first geometry optimisation step. The result from the first SCF cycle affects the convergence of subsequent steps and with the better starting point the time taken to complete the geometry relaxation was reduced and the chance of success was increased.

O.0.5 Not Straying Too Far From Reality

It is important to provide the calculation with reasonable and realistic configurations and settings. Where a pure element such as Aluminium, Iron or Platinum was being investigated, the known experimental lattice parameters were used as a starting point (table 7.1 and table 7.2). If the parameters were too far from these values, for example inputting a lattice parameter of either 2 or 8 angstroms for Aluminium rather than 4, as a starting approximation, the DFT calculation might struggle to relax the structure, or to even successfully run or converge a calculation.

The FCC Iron structure lattice parameter was estimated from the density of FCC steel, and the lattice parameter for alloys could similarly be estimated within a reasonable margin to use as a start point for the calculation.

O.0.6 Trial Parameters for Iron with and Octahedral Defect

Calculations where there is a high level of symmetry finished relatively quickly and without much trouble. Where the position of each atom was perturbed slightly or where there was a defect the SCF calculations would take more iterations to converge and would often fail completely. Where the geometry and atom position was

also being optimised through repeated sets of SCF iterations the calculation would begin to converge but would ultimately fail through SCF non convergence.

Nbnd	Ntype	K	Degauss	Scfconv	Beta	Ndim	Mode	Iterations
317	5	9x9x9	0.04	1.0×10^{-6}	0.1	10	Plain	75
320	3	9x9x9	0.04	1.0×10^{-6}	0.05	10	Local-TF	46
317	3	9x9x9	0.04	1.0×10^{-7}	0.1	10	Local-TF	42
340	3	9x9x9	0.04	1.0×10^{-6}	0.8	8	Local-TF	38
320	3	9x9x9	0.04	1.0×10^{-6}	0.1	10	Local-TF	36
317	3	9x9x9	0.04	1.0×10^{-6}	0.8	8	Local-TF	33
317	5	9x9x9	0.04	1.0×10^{-6}	0.3	8	Local-TF	32
317	3	9x9x9	0.04	1.0×10^{-6}	0.3	8	Local-TF	32
320	3	9x9x9	0.04	1.0×10^{-6}	0.3	10	Local-TF	29
320	3	11x11x11	0.04	1.0×10^{-6}	0.3	10	Local-TF	29
320	3	13x13x13	0.04	1.0×10^{-6}	0.3	10	Local-TF	29
320	3	9x9x9	0.07	1.0×10^{-6}	0.3	10	Local-TF	25

Table O.1: Improving convergence SCF calculations of defects in Iron

Iron with an octahedral defect was used to help fine tune parameters. Initially a set of SCF calculations were performed with different parameters, and finally these were used to relax the cell geometry and atom positions for this defect.

Changing the number of k-points did not have an impact on the number of iterations required to converge, but changing the smearing value did. An increase in smearing would be detrimental to the accuracy of convergence to the correct value and by doing this it would cause a difference in energy between past calculations with a smearing of 0.04Ry and those with 0.07Ry.

The adjustment that had the largest impact on reducing the number of iterations was the change from plain mixing to Local-TF. Following this, using a mixing beta of 0.3 where more weight is given to the previous state, also helped to reduce the number of iterations.

The attempts to optimise the values was not exhaustive as there were constrictions on time. However, despite making these improvements and reducing the convergence threshold to 1.0×10^{-8} , the cell geometry and position relaxation of the starting configuration for an octahedral defect (by adding an atom at the octahedral location) also fails to relax by SCF non-convergence within an iteration of the relaxation algorithm.

The optimization of the geometry and atom positions was likely due to it being a complex system to model with the initial configuration too far from the relaxed one. To work around this the octahedral defect for a more simple metal, Aluminium, was relaxed to give a better starting point for the Iron calculation. The increase in cell size was also applied. With a better initial configuration and the adjusted parameters, the Iron octahedral calculation successfully completed.

This process was repeated for tetrahedral, 100, 110, 111 and crowdion defects.

Appendix P

Transferability of Interatomic Potentials

P.1 Sheng Aluminium Potential

P.1.1 FCC Structure

The EAMPA code was used to compute both the Birch-Murnaghan and Rose-Vinet equations of state using the known values for Aluminium and the Sheng Al potential[43].

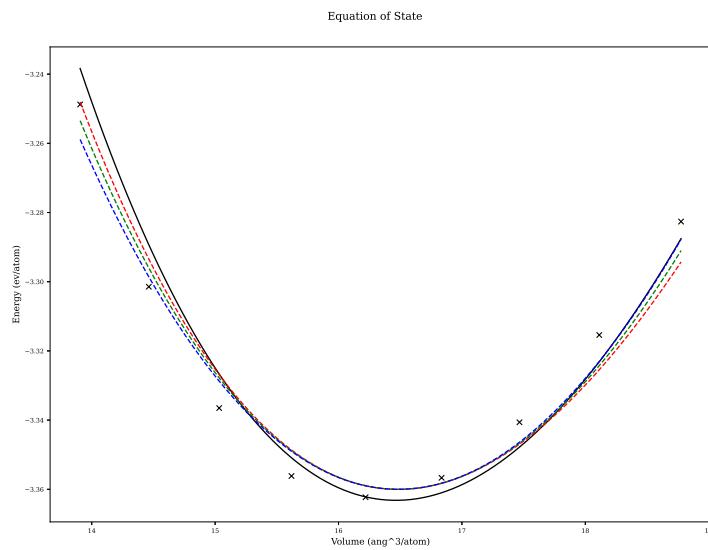


Figure P.1: Birch Murnaghan equation of state for FCC Aluminium - comparison of experimental data to the Sheng Al potential

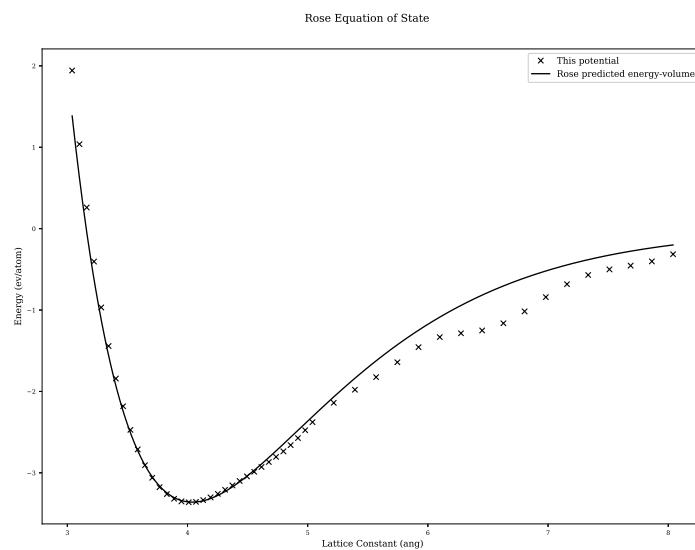


Figure P.2: Rose-Vinet equation of state for FCC Aluminium - comparison of experimental data to the Sheng Al potential

P.1.2 BCC Structure

P.2 Mendelev et al 2003 Iron Potential

The EAMPA code was used to compute both the Birch-Murnaghan and Rose-Vinet equations of state using the known values for Aluminium and the Sheng Al potential[143].

P.2.1 BCC Structure

P.2.2 FCC Structure

P.3 Ackland et al 1997 Iron Potential

The EAMPA code was used to compute both the Birch-Murnaghan and Rose-Vinet equations of state using the known values for Aluminium and the Sheng Al potential[**Fe·abch**].

P.3.1 BCC Structure

P.3.2 FCC Structure

The EAMPA code was used to compute both the Birch-Murnaghan and Rose-Vinet equations of state using the known values for Aluminium and the Sheng Al potential[43].

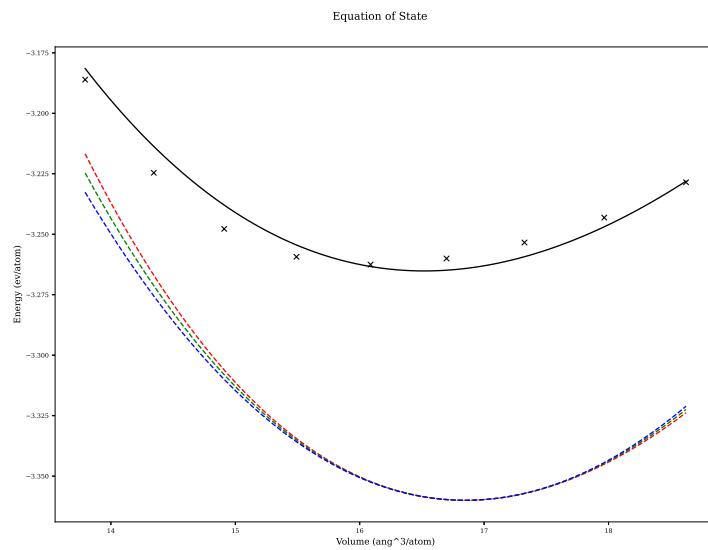


Figure P.3: Birch Murnaghan equation of state for BCC Aluminium - comparison of experimental data to the Sheng Al potential

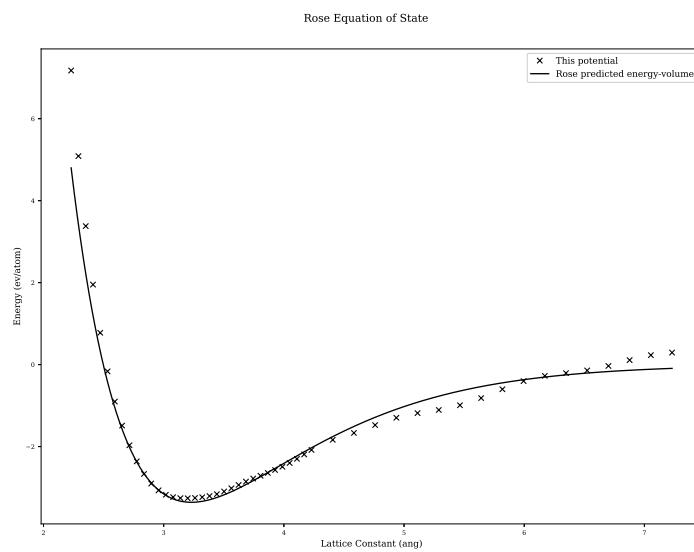


Figure P.4: Rose-Vinet equation of state for BCC Aluminium - comparison of experimental data to the Sheng Al potential

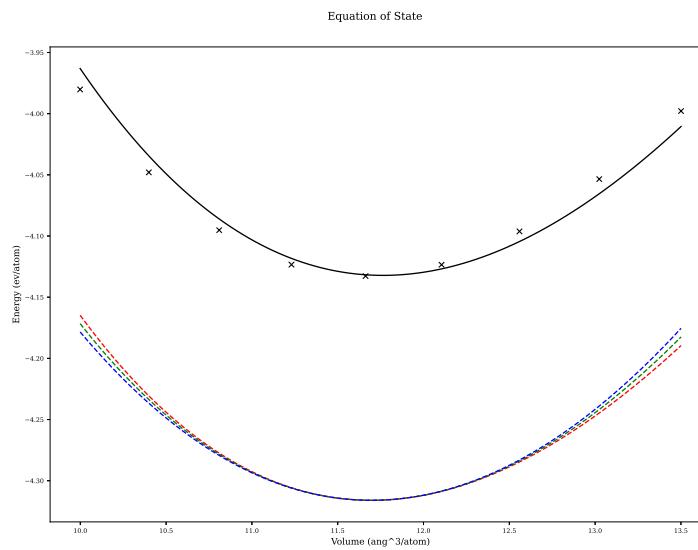


Figure P.5: Birch Murnaghan equation of state for FCC Aluminium - comparison of experimental data to the Sheng Al potential

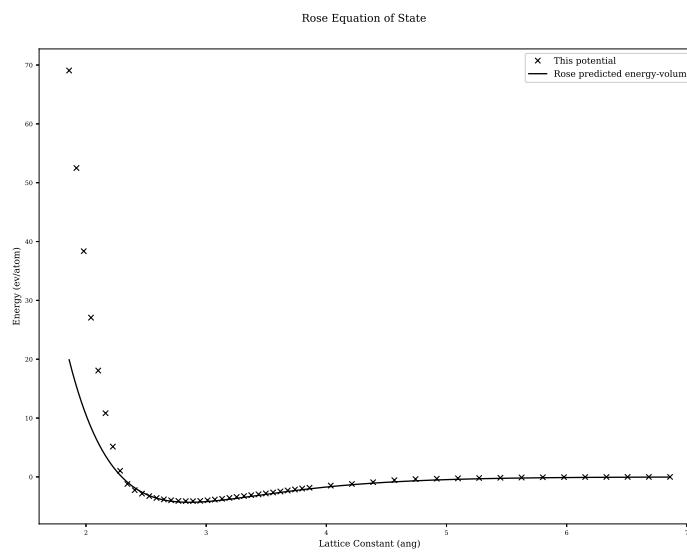


Figure P.6: Rose-Vinet equation of state for FCC Aluminium - comparison of experimental data to the Sheng Al potential

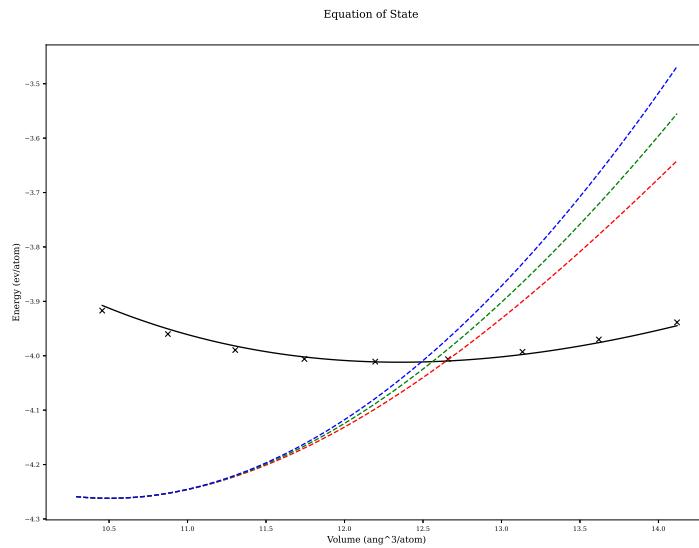


Figure P.7: Birch Murnaghan equation of state for FCC Aluminium - comparison of experimental data to the Sheng Al potential

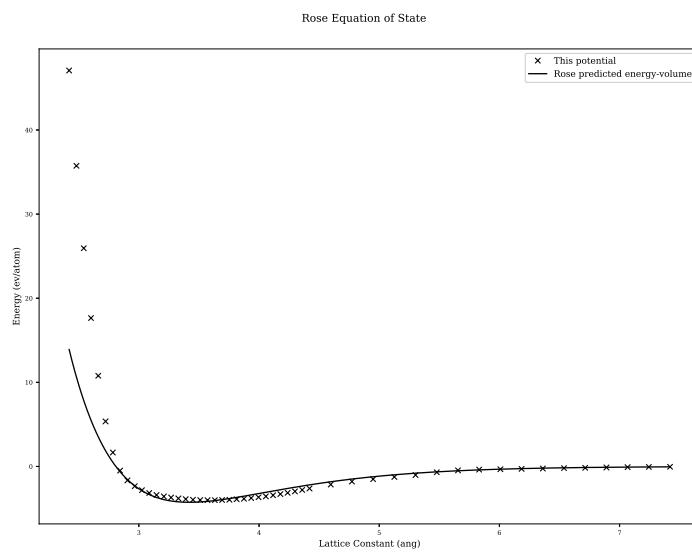


Figure P.8: Rose-Vinet equation of state for FCC Aluminium - comparison of experimental data to the Sheng Al potential

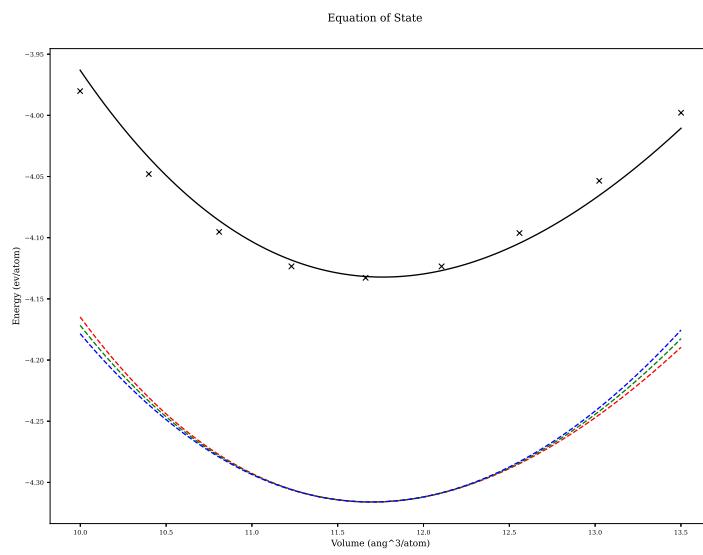


Figure P.9: Birch Murnaghan equation of state for FCC Aluminium - comparison of experimental data to the Sheng Al potential

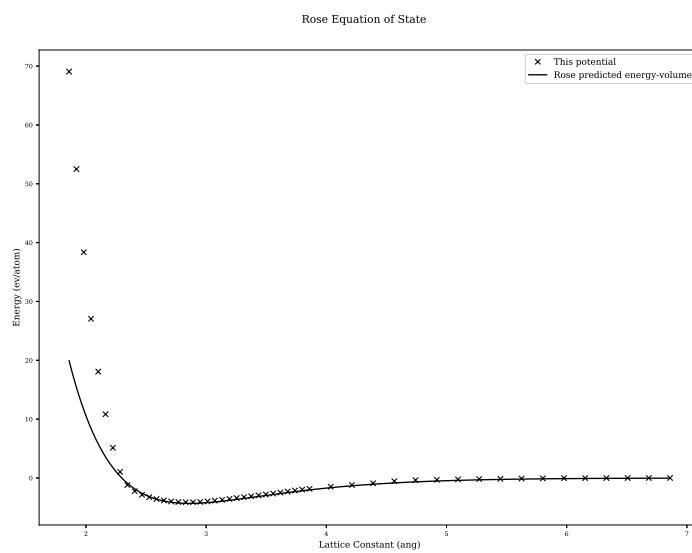


Figure P.10: Rose-Vinet equation of state for FCC Aluminium - comparison of experimental data to the Sheng Al potential

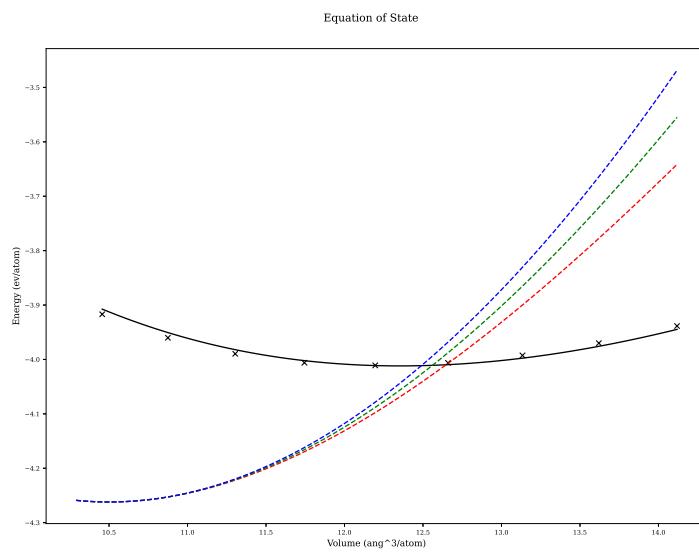


Figure P.11: Birch Murnaghan equation of state for FCC Aluminium - comparison of experimental data to the Sheng Al potential

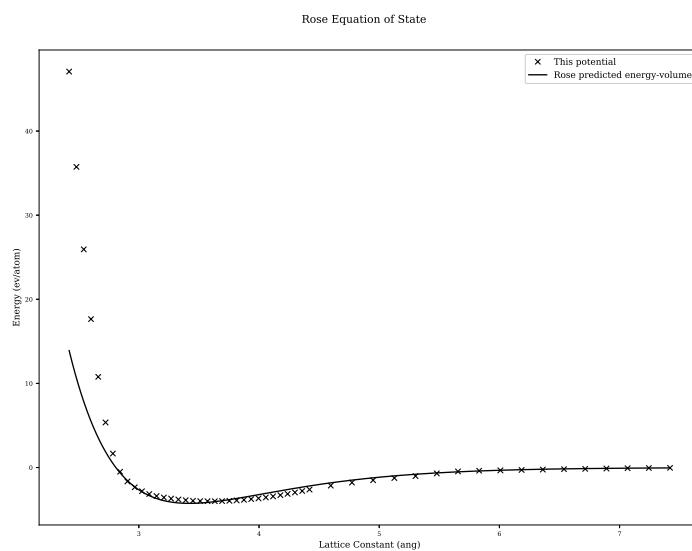


Figure P.12: Rose-Vinet equation of state for FCC Aluminium - comparison of experimental data to the Sheng Al potential

Appendix Q

Optimization

Q.1 Optimization

Q.1.1 Introduction

Optimization is the process of finding the best solution for a problem that may also need to satisfy a number of imposed constraints. In terms of this work, there are several tasks where optimization plays a key role. The DFT code used, PWscf, employs the Broyden Fletcher Goldfarb Shanno algorithm to relax structures to give the optimum volume and optimal structure of an arrangement of atoms. During the fitting of the variety options of equation of state to the energy-volume points, the Levenberg-Marquardt Algorithm is used. Finally, a genetic algorithm and simulated annealing algorithm are coupled with LMA to locate the global optimal parameters for interatomic potential functions.

Q.1.2 Continuous and Discrete Optimization

Problems that may be solved using discrete optimisation have, as the name suggests, discrete solutions. An example would be the travelling salesman problem, where the aim is to minimise the distance travelled by the salesman between cities. An example of the solution may be city E to A to B to C to D. There's a set of possible solutions to the problem. If the problem was changed such that any point within a 1 mile radius of the centre of each city is a valid solution, and the aim is to find the optimal points to travel between within the area of each city, it would become a continuous problem.

Fitting interatomic potentials to data is an example of continuous optimisation. The Morse potential takes three parameters plus the separation r between the two atoms. If the parameters are being optimised such that Morse potential matches the experimental forces (assuming they could be measured experimentally) between two atoms for a set of separation values, the choice of parameters isn't discrete; they are taken from a continuous range of real numbers.

Q.1.3 Constrained and Unconstrained Optimization

There are constraints that may need to be considered, and constraints that apply to the fitting of interatomic potentials are as follows:

- continuous well behaved function
- continuous first-derivative, smooth change in force wrt r

- positive electron density
- smooth cutoff at some $r = r_{cut}$
- repulsion (ZBL/exponential) as r approaches 0
- continuous second-derivative
- enough parameters coupled with a good choice of functions so the potential is flexible enough to replicate all the experimental data well
- a small enough range of parameters to keep the parameter space small enough to search in a reasonable amount of time
- maintain physical elegance
- ignore physical elegance

Certain constraints listed will battle one-another, while others may be dropped, such as restricting to positive electron-densities if the choice to ignore physical elegance is selected.

Q.1.4 Global and Local Optimization

An example to discuss the difference between local and global optimization will be given in a very literal sense. I am located near Birmingham, and I wish to find the highest point local to me. Using a modern smartphone and map I could quickly find the nearest hill to my current position; this would be the local maxima. I would, however, need to go up and down many peaks and troughs, scouring the entire surface of the Earth, until I found the global maxima, somewhere amongst the many peaks and troughs in the Himalayas.

When optimising the parameters of a function, it is relatively easy to find a local extreme that would give me the optimum parameters locally. It is a much harder task to find the global optimum, especially if very little, or nothing at all, is known about the function for which the parameters are being optimised.

Q.1.5 Global Optimization Algorithms

Simulated Annealing

A large amount of computing power is required today to solve many problems in Physics. The optimum arrangement of Iron atoms, at various concentrations and temperatures is one such example, but nature knows the correct structure it should take because it has access to the rule book. It makes perfect sense to look to nature when designing optimisation algorithms.

Simulated annealing mimics the way a solid cools, with its atoms settling into the optimum relaxed positions as time passes. Initially, the solid is hot and the atoms are able to take non-optimal positions, but as the solid cools the atoms take their optimal places. The heating allows the atoms to move from non-optimal positions in the initial configuration.

The algorithm was originally developed to improve upon the Metropolis algorithm, solving the issue of becoming stuck in non-global minima. Motivation for the algorithm included applying it to the optimum placement of chips on a circuit board[227], the travelling salesman (and similar) problem as well as finding the optimum arrangement of atoms.

Listing Q.1: Pseudocode for the simulated annealing algorithm

```

1 # Calculate the starting rss
2 rss = get_rss(p)
3
4 # Loop (n times, until a threshold is met, time runs out etc)
5 while(loop)
6 {
7     p_new = vary(p)           # Make a new set of parameters
8     rss_new = get_rss(p_new) # Get rss for these parameters
9
10    # Always accept BETTER
11    # Sometimes accept WORSE (depends on temperature and how bad it is)
12    if(rss_new < rss or rand(0.0,1.0) < exp((rss-rss_new) / temp))
13    {
14        p = p_new
15        rss = rss_new
16    }
17 }
```

The simulated annealing algorithm takes in a starting set of parameters. These are varied and, if the new parameters give a better solution to the problem, the parameters are updated. However, there is a chance that a worse set of parameters are used, and this depends on how bad they are and the current value of the temperature in the algorithm (listing Q.1). The algorithm used is this work is available in the appendix (section S.5).

Genetic Algorithm

It is unclear whether life started on this planet, or was brought to this planet, and this idea only removes the problem of how life as we are able to understand it was conceived to a different time and place. However it started, there is evidence of life on this planet approximately 3.5 billion years ago. Prokaryotes were the simple single cell organisms that inhabited our planet for billions of these. Their cell does not have a nucleus and the genetic information is contained in RNA and DNA within the cell.

Approximately 1.5 billion years ago, life made a leap forward in complexity and evolved into the first Eukaryotic cells. These are larger and more complex than Prokaryotes, and the genetic material is stored within a nucleus. Through processes of inheritance, variation, natural selection and the vast expanse of time, more complex organisms developed.

A set of solutions are bred with one another, and where there is an improvement the new variation is reinserted into the gene pool. There is a danger that the population can become a set of clones, so a large pool of sets of parameters is required, fresh parameters proposed every so often as well as a mechanism to prevent clones appearing.

As simulated annealing takes inspiration from cooling solids, genetic algorithms take inspiration from evolution. The algorithm used is this work is available in the appendix (section S.5).

Basin-Hopping

Lennard-Jones clusters are test systems of atoms models using the potential of the same name. Clusters of varying sizes have an optimal configuration with the minimum amount of energy. An algorithm was developed in the 1990s to improve upon existing global search algorithms and was applied to clusters of between 2 and 110 atoms[228]. The resulting basin-hopping algorithm allowed Wales and Doye to find three previously unknown minima.

$$\tilde{E}(\vec{X}) = \min(E(\vec{x})) \quad (\text{Q.1})$$

The objective of the algorithm was to solve eq. Q.1 where the vector \vec{X} is the coordinates of the atoms. A number of iterations, set by the user, are stepped through. The co-ordinates are randomly perturbed before the local minima is determined by using a local optimizer such as BFGS, LM-BFGS or CG[229]. Similar to SA, the minimized parameters are either accepted or discarded dependent upon how good the new parameters are.

Q.1.6 Local Optimization Algorithms

Quadratic Programming (QP)

The method of Quadratic Programming, or quadratic optimization, is the task of minimising the matrix equation eq. Q.2 subject to imposed conditions[144].

$$\begin{aligned} \min_x q(\vec{x}) &= \frac{1}{2} \vec{x}^T \vec{G} \vec{x} + \vec{x}^T \vec{c} \\ \text{subject to } &\vec{a}_i^T \vec{x} = b_i, i \in \varepsilon \\ &\vec{a}_i^T \vec{x} \geq b_i, i \in I \end{aligned} \quad (\text{Q.2})$$

This is the method used by Bonny et al. in their potential fitting program and will be discussed in section 5.6.

Gradient Descent

The gradient descent method is a reliable way of searching for a local minimum. It requires only the first derivative coupled with a line search to move closer to the minimum point. Close to the minimum, it is slower than the Newton-Gauss and LMA, but these require the computation of the Jacobian to estimate the Hessian matrix, so these are more computationally intensive. A pseudocode for the algorithm is available in the appendix (section S.5).

Newton-Gauss

The Newton-Gauss method is an algorithm that finds the local minimum of a function by approximating the Hessian matrix with the Jacobian and its transpose (eq. Q.3). By estimating the Hessian, the algorithm only requires the computation of the first order derivatives with respect to each parameter, at each point pair x , y . It is possible to use the function and the analytical derivative, but as this work requires the calculation of functions that have no simple analytical form, the first derivative will be approximated within the algorithm.

$$\begin{aligned} \vec{J}^T \vec{J} \vec{p} &= \vec{J}^T \vec{r} \\ \vec{r} &= \vec{y} - f(\vec{x}, \vec{p}) \end{aligned} \quad (\text{Q.3})$$

Close to a minima, the NG method quickly moves to the optimum point. Depending on the starting point, it may converge only to a local minimum, and not the global, and it may also be unstable, not converging at all.

Levenberg Marquardt

The LMA is a NG type algorithm that includes the addition of a dampening term that helps to increase the robustness of the algorithm. The particular algorithm here uses a number of other schemes to improve the overall effectiveness of the algorithm. The equation at the center of the algorithm is similar to that of NG (eq. Q.4).

$$\begin{aligned} \left(\vec{J}^T \vec{W} \vec{J} + \lambda \text{diag}(\vec{J}^T \vec{W} \vec{J}) \right) \vec{p} &= \left(\vec{J}^T \vec{W} \vec{r} \right) \\ \vec{r} &= \vec{y} - f(\vec{x}, \vec{p}) \end{aligned} \quad (\text{Q.4})$$

The starting value for lambda is calculated as a function of the estimate of the Hessian and a cutoff for lambda is introduced to remove dampening altogether. A delayed gratification scheme has also been introduced to increase lambda by 50 percent if the trial solution is worse, and to decrease lambda by a factor of 5 if the trial solution is better. Finally, a diagonal weighting matrix has been included to allow certain parameters to be preferential during their optimisation.

Conjugate Gradient

In the method of steepest decent an initial value is selected and the gradient at that point is computed. This gives a search direction to look in and, following a line search, a new point is selected where the gradient is orthogonal to the search gradient. The process is iterative and at each new point the search gradient is replaced with the new orthogonal gradient (eq. Q.5[230]).

$$\begin{aligned} r_{(i)} &= b - Ax_{(i)} \\ \alpha &= \frac{r_i^T r_{(i)}}{r_{(i)}^T A r_{(i)}} \\ x_{(i+1)} &= x_{(i)} + \alpha_{(i)} r_{(i)} \end{aligned} \quad (\text{Q.5})$$

The conjugate gradient method is similar.

$$\begin{aligned} d_{(0)} &= r_{(0)} = b - Ax_{(0)} \\ \alpha_i &= \frac{r_{(i)}^T r_{(i)}}{d_{(i)}^T A d_{(i)}} \\ x_{(i+1)} &= x_{(i)} + \alpha_{(i)} d_{(i)} \\ r_{(i+1)} &= r_{(i)} - \alpha_{(i)} A d_{(i)} \\ \beta_{(i+1)} &= \frac{r_{(i+1)}^T r_{(i+1)}}{r_{(i)}^T r_{(i)}} \\ d_{(i+1)} &= r_{(i+1)} + \beta_{(i+1)} d_{(i)} \end{aligned} \quad (\text{Q.6})$$

Broyden Fletcher Goldfarb Shanno

The BFGS method is a Quazi-Newton method where the Hessian is approximated in eq. Q.7[144].

$$\vec{B}_{k+1} = \vec{B}_k - \frac{B_k S_k S_k^T B_k}{S_k^T B_k S_k} + \frac{y_k y_k^T}{y_k^T s_k} \quad (\text{Q.7})$$

This optimization method is implemented in SciPy in Python along with a number of other methods listed here.

Python SciPy Optimization Functions

Many optimization functions are available in the EAMPA code through SciPy. This is a library that is well used in the Scientific community and it is easy to implement.

Appendix R

Interatomic Potential: Energy, Force Stress Calculations

R.0.1 Calculating Energy, Force and Stress

Molecular dynamics simulations do not need to include the weak or strong force, and the gravitational forces between atoms in the simulated material are so weak in comparison to the electromagnetic force, they can also be ignored. There is a force between all the atoms within a real material, but the electromagnetic force is inversely proportionally to the separation of the atoms. Above a certain separation, the electromagnetic force can also be ignored, in order to simplify the computation.

Neighbour List

A cut off radius can be introduced to limit the number of neighbours (section 5.3.4). As the lattice parameter decreases, the atoms are brought closer together, and as the cut off radius is increased more atoms are considered to be within the sphere of influence of one another. Both increase the number of neighbours each atom has (fig. R.1).

Building a neighbour list may take a long time, depending on the parameters. If a simplistic approach is taken, for N atoms in the supercell, there will be approximately $27N^2$ checks between atoms to see whether they are within the cut off radius of one another. For larger numbers of atoms in a supercell, the whole may be decomposed into smaller domains.

For example, a 16x16x16 FCC supercell, containing 16,384 atoms, would require looping $27 \times 16,384^2 = 7.25 \times 10^9$ times. Breaking the supercell into 64 smaller domains with 256 atoms in each, reducing the problem to $64 \times 27 \times 256^2 = 1.13 \times 10^8$ loops.

For this work, the supercells used to calculate bulk properties from the interatomic potentials, and the configurations generated by DFT, contain fewer than 1,000 atoms, removing the need to decompose the configuration into smaller sub cells.

To build the neighbour list a halo configuration is created such that it lies on top of the real atoms, but extends at least as far as the cut off radius on each side of the real simulation box (fig. R.2). Periodic boundary conditions are used to construct the halo.

Once the halo list is computed, the neighbour list may also be computed by looping through the real atoms and halo atoms. A simple pseudo coded sub routine is given (listing R.1).

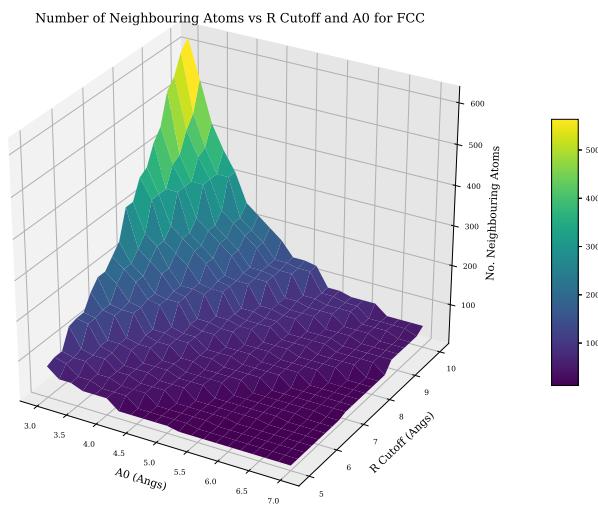


Figure R.1: Neighbouring atom count dependant upon lattice parameter and cutoff radius

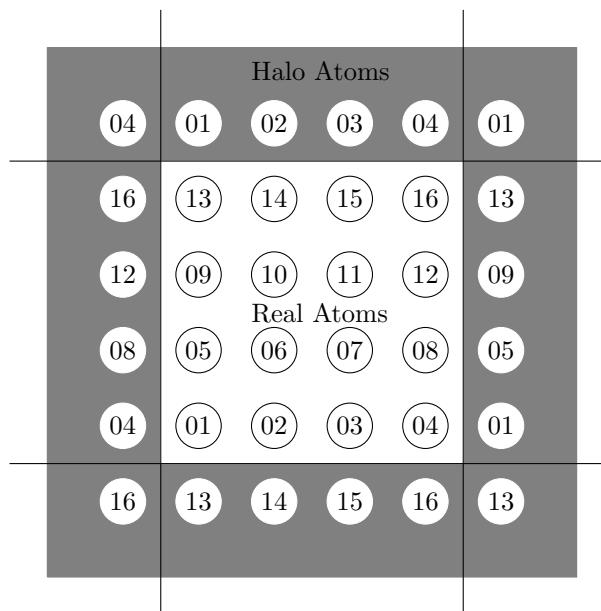


Figure R.2: A 2D representation of atoms in simulation box with a halo of atoms at the boundary

Listing R.1: Simple subroutine for generating a half size neighbour list

```

1
2 ! real_atoms - array holding coordinates of all real atoms
3 ! halo_atoms - array holding coordinates of all halo atoms
4 ! real_ids - unique id for each atom
5 ! halo_ids - unique atom ids for halo atoms
6 ! nlist_ids - array to store ids
7 ! nlist_r - array to store separation
8
9 ! Define the cutoff and counter start value
10 r_cut = 5.0
11 nl_counter = 1
12
13 ! Calculate square
14 r_cut_sq = r_cut ** 2
15
16 ! Loop over all real atoms
17 DO n_real = 1, real_atom_count
18   ! Loop over all halo atoms
19   DO n_halo = 1, halo_atom_count
20     IF (real_ids(n_real) .LT. halo_ids(n_halo)) THEN
21       r(1:3) = halo_atoms(n_halo, :) - real_atoms(n_real, :)
22       r_sq = SUM(r(1:3) * r(1:3))
23       IF (r_sq .LE. r_cut_sq) THEN
24         r_mag = sqrt(r_sq)
25         nlist_ids(nl_counter, 1) = real_ids(n_real)
26         nlist_ids(nl_counter, 2) = halo_ids(n_halo)
27         nlist_r(nl_counter, 1) = r_mag
28         nlist_r(nl_counter, 2:4) = r(1:3)/r_mag
29         nl_counter = nl_counter + 1
30       END IF
31     END IF
32   END DO
33 END DO

```

The resulting list will contain unique pair combinations; i.e. the pair atom 1 and atom 2 will only be recorded once, and not also as atom 2 and atom 1.

Many modern MD codes are designed to run on computers with many thousands of processor cores. This leads to a choice of approaches, either to keep the configuration in shared memory and allow multiple processes to work on the same data, or break the data up, distributing it across various nodes.

In the case of DL_POLY a link-cell domain decomposition is used[231]. The supercell is divided up into subcells, and these must be at least the size of the cut-off radius. Each processor will work on a geometric domain and these will only interact with their immediate neighbours. The halo data for each subcell is only passed between processors before and after the calculation of the equations of motion of the atoms within each subcell.

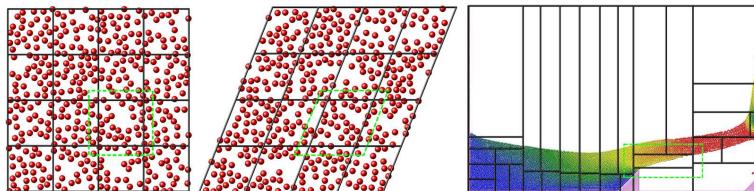


Figure R.3: Partitioning in LAMMPS[232]

This is also the approach used by the LAMMPS code. The simulation is partitioned into either boxes where the density of atoms is uniform or tiles where there are regions with few or no atoms (fig. R.3). This ensures processors are evenly utilised.

Computing Total Energy

The total energy of the system is the sum of the individual energies for the atoms in the simulation. The type of atom (or pairs of atoms) will determine which function is used.

First, to compute the pair potentials:

- set the total energy of the system equal to zero
- set the starting energy for each atom in the system to zero
- loop through the atom pairs in the entire neighbour list
- for each atom pair, A and B, use the known separation and the potential function to compute the potential energy on atom A due to B and vice versa
- add this potential energy to both A and B
- after looping through the neighbour list, add all the energies due to the pair potential to the total energy

For an EAM or 2BEAM potential, the densities and embedding energies must also be computed:

- set the electron density at the position of (for each) atom to zero
- loop through the atom pairs in the entire neighbour list
- for each atom pair, using the density function for the atom A, the density at atom B due to atom A will be calculated and added to the density at atom B
- if the atoms are both of the same type the same density will be added to atom A due to atom B
- if the atom types are different, using the density function for the atom B, the density at atom A due to atom B will be calculated and added to the density at atom A
- following looping through the neighbour list, to calculate the densities at each atom, the list of atoms will be looped through
- for each atom, the density value at the position of that atom, will be input into the appropriate embedding function to calculate the embedding portion of the energy
- add all the embedding energies to the total energy of the system

For a 2BEAM potential, repeat the above procedure for the second group of density functions and embedding functions (do not repeat calculation of the pair potentials).

Computing Forces on Atoms

In order to calculate the forces on the atoms with an EAM or 2BEAM potential, the neighbour list and atom list will need to be looped through several times. First, the pair potential and force due to the pair potential must be calculated by a complete loop through the neighbour list. At the same time, the density at each atom location is also calculated. The embedding energy may then be computed by looping through all the atoms and using the electron density at each atom to give the energy of the atom embedded in that density. The third and final loop will run through the neighbour pairs in the neighbour list once more computing the force on each atom due to its embedding in the electron background.

$$\begin{aligned}
F_{pair}^k &= - \sum_{j=1, j \neq k}^N \frac{\partial V_{kj}(r_{kj})}{\partial r_{kj}} \frac{\vec{r}_{kj}}{|\vec{r}_{kj}|} \\
F_{embed}^k &= - \sum_{j=1, j \neq k}^N \left(\frac{\partial F}{\partial \rho_k} \frac{\partial \rho(r_j)}{\partial r_{kj}} + \frac{\partial F}{\partial \rho_j} \frac{\partial \rho(r_k)}{\partial r_{kj}} \right) \frac{\vec{r}_{kj}}{r_{kj}}
\end{aligned} \tag{R.1}$$

The force on each atom, due to surrounding atoms, may be split into two; force due to the pair potential, and the force due to the embedding of the atom (eq. R.1)[136][231].

Computing Stress

The stress is to be calculated assuming the system is at 0K[233] so the virial stress equation is eq. R.2.

$$\tau_{ij} = \frac{1}{2V} \sum_{k, l \in V} \left(x_i^{(l)} - x_i^{(k)} \right) f_j^{(kl)} \tag{R.2}$$

Within the computer code, the individual force components between pairs of atoms are stored as well as an equal sized array of vectors representing their spatial separation. Those pairs with one atom in the halo around the simulation box are used to calculate the stress on the simulation box.

Pseudo Code for the Energy, Force and Stress Subroutine

The calculation requires several arrays to temporarily store data. The electron densities and gradients of the density function are stored for multiple groups of densities and embedding functionals. The overall forces on each atoms also need to be stored as well as the force between pairs of atoms, as this is required to compute the virial stress. The Fortran code to compute this is included in the appendix (section S.1) and a pseudocode in listing R.2 explains the process.

Listing R.2: Pseudo Code for Energy and Stress Force Calculation

```

1
2 electron_density[1:n_atoms, 1:bands] = 0.0 ! electron density at each atom
3 energy = 0.0                                ! total energy
4 forces[:, :] = 0.0                           ! forces each atom, 3D
5 force_between_pairs[:, :] = 0.0              ! used to store force between pairs, for stress computation
6 stress[:, :] = 0.0D0

7
8 ! LOOP 1 - Pair energy, force and calculate densities
9 DO n = 1, neighbour_count
10    E = get_PairEnergy(atom_a, atom_b, r[n,:])
11    F[:] = get_PairForce(atom_a, atom_b, r[n,:])

12
13    ! Save energy
14    energy = energy + E

15
16    ! Save Force
17    f[atom_a, :] = f[atom_a, :] - F[:]
18    f[atom_b, :] = f[atom_b, :] + F[:]
19    force_between_pairs[:, :] = force_between_pairs[:, :] - F[:] // Used to calculate stress

20
21    ! Loop through density bands
22    DO band = 1, bands
23        ! Electron density at A due to atom B

```

```
24     electron_density[atom_a, band] = get_Density(atom_b, r)
25
26     ! Electron density at B due to atom A
27     electron_density[atom_b, band] = get_Density(atom_a, r)
28 END DO
29
30
31 ! LOOP 2 - Embedding energy
32 DO n = 1, atom_count
33     ! Loop through density bands
34     DO band = 1, bands
35         energy = energy + get_EMBEDDINGEnergy(n, band)
36     END DO
37 END DO
38
39 ! LOOP 3 - Embedding force
40 DO n = 1, neighbour_count
41     ! Loop through density bands
42     DO band = 1, bands
43         dgradA = get_DensityGradient(atom_b, r, band)
44         dgradB = get_DensityGradient(atom_a, r, band)
45         fgradA = get_EMBEDDINGGradient(atom_a, electron_density[atom_a, band])
46         fgradB = get_EMBEDDINGGradient(atom_b, electron_density[atom_b, band])
47
48         F[:] = (fgradA * dgradB + fgradB * dgradA) * r[n, :]
49
50         f[atom_a, :] = f(atom_a, :) - F[:]
51         f(atom_a, :) = f(atom_a, :) + F[:]
52
53         force_between_pairs[n,:] = force_between_pairs[n,:] - F[:] // Used to calculate stress
54     END DO
55 END DO
56
57 ! LOOP 4 STRESS
58 DO n = 1, neighbour_count
59     ! Only compute if the second atom is in the halo
60     IF(nlisthalo(n))THEN
61         DO i = 1,3
62             DO j = 1,3
63                 stress[i,j] = stress[i,j] + (r[n, i] * force_between_pairs[n,j])
64             END DO
65         END DO
66     END IF
67 END DO
68 stress[1:3, 1:3] = stress[1:3, 1:3] / (2.0 * volume)
```

Appendix S

EAMPA Code

The full code is available at <https://github.com/BenPalmer1983/eampa>

Several of the important functions and subroutines that make up the code are listed below.

S.1 Energy, Force, and Stress Subroutine

Listing S.1: Energy, Force and Stress subroutine (Fortran)

```
1 SUBROUTINE efs(nlabels, r, rvec, inhalo, clabels, volume, energy, forces, stress)
2 !#####
3 IMPLICIT NONE
4 !#####
5 INTEGER(kind=StandardInteger), INTENT(IN) :: nlabels(:, :)
6 REAL(kind=DoubleReal), INTENT(IN) :: r(:)
7 REAL(kind=DoubleReal), INTENT(IN) :: rvec(:, :)
8 INTEGER(KIND=StandardInteger), INTENT(IN) :: inhalo(:)
9 INTEGER(KIND=StandardInteger), INTENT(IN) :: clabels(:)
10 REAL(kind=DoubleReal), INTENT(IN) :: volume
11 REAL(kind=DoubleReal), INTENT(INOUT) :: energy(1:3)
12 REAL(kind=DoubleReal), INTENT(INOUT) :: forces(:, :)
13 REAL(kind=DoubleReal), INTENT(INOUT) :: stress(:, :)
14 !#####
15 INTEGER(KIND=StandardInteger) :: n, cc, nc, fn, fgroup
16 INTEGER(KIND=StandardInteger) :: i, j
17 REAL(kind=DoubleReal), ALLOCATABLE :: density(:, :)
18 REAL(kind=DoubleReal), ALLOCATABLE :: nl_force(:, :)
19 REAL(kind=DoubleReal), ALLOCATABLE :: fgradA(:)
20 REAL(kind=DoubleReal), ALLOCATABLE :: fgradB(:)
21 REAL(kind=DoubleReal), ALLOCATABLE :: dgradA(:)
22 REAL(kind=DoubleReal), ALLOCATABLE :: dgradB(:)
23 REAL(kind=DoubleReal), ALLOCATABLE :: density_thread(:, :)
24 REAL(kind=DoubleReal), ALLOCATABLE :: nl_force_thread(:, :)
25 REAL(kind=DoubleReal) :: energy_thread(1:3)
26 REAL(kind=DoubleReal) :: forces_thread(1:SIZE(forces,1), 1:SIZE(forces,2))
27 REAL(kind=DoubleReal) :: y(1:2)
28 REAL(kind=DoubleReal) :: fv(1:3)
29 !#####
30 ! Neighbourlist Count
31 nc = SIZE(nlabels, 1)
32 cc = SIZE(clabels, 1)
33
34 ! Allocate density array
```

```

35 ALLOCATE(density(1:cc, 0:group_max-1))
36 ALLOCATE(density_thread(1:cc, 0:group_max-1))
37 ALLOCATE(fgradA(0:group_max-1))
38 ALLOCATE(fgradB(0:group_max-1))
39 ALLOCATE(dgradA(0:group_max-1))
40 ALLOCATE(dgradB(0:group_max-1))
41 ALLOCATE(nl_force(1:nc, 1:3))
42 ALLOCATE(nl_force_thread(1:nc, 1:3))

43
44 ! Zero
45 energy = 0.0D0
46 density = 0.0D0
47 forces = 0.0D0
48 stress = 0.0D0
49 nl_force = 0.0D0
50
51 !$OMP PARALLEL &
52 !$OMP PRIVATE(n, y, fgroup, fn, energy_thread, density_thread, &
53 !$OMP forces_thread, fv, nl_force_thread) &
54 !$OMP SHARED(nc, nlabels, r, rvec, energy, density, forces, nl_force)
55 energy_thread(1) = 0.0D0
56 density_thread(:, :) = 0.0D0
57 forces_thread(:, :) = 0.0D0
58 nl_force_thread(:, :) = 0.0D0
59
60 !$OMP DO
61 DO n = 1, nc
62 ! PAIR ENERGY
63 CALL getparr(1, nlabels(n,1), nlabels(n,2), 0, r(n), y)
64 energy_thread(1) = energy_thread(1) + y(1)
65 ! PAIR FORCE
66 fv(:) = y(2) * rvec(n, 1:3)
67 forces_thread(nlabels(n, 3), 1:3) = forces_thread(nlabels(n, 3), 1:3) + fv(1:3)
68 forces_thread(nlabels(n, 4), 1:3) = forces_thread(nlabels(n, 4), 1:3) - fv(1:3)
69 nl_force_thread(n, 1:3) = -fv(1:3)
70
71 ! DENSITY
72 IF(nlabels(n,1) .EQ. nlabels(n,2))THEN
73   fn = 0
74   DO WHILE(dens_index_a(nlabels(n,1), fn) .GT. -1)
75     fgroup = dens_index_a(nlabels(n,1), fn)
76     CALL getp(2, nlabels(n,1), -1, fgroup, r(n), y(1))
77     density_thread(nlabels(n,4), fgroup) = density_thread(nlabels(n,4), fgroup) + y(1)
78     density_thread(nlabels(n,3), fgroup) = density_thread(nlabels(n,3), fgroup) + y(1)
79     fn = fn + 1
80   END DO
81 ELSE
82   fn = 0
83   DO WHILE(dens_index_a(nlabels(n,1), fn) .GT. -1)
84     fgroup = dens_index_a(nlabels(n,1), fn)
85     CALL getp(2, nlabels(n,1), -1, fgroup, r(n), y(1))
86     density_thread(nlabels(n,4), fgroup) = density_thread(nlabels(n,4), fgroup) + y(1)
87     fn = fn + 1
88   END DO
89   fn = 0
90   DO WHILE(dens_index_a(nlabels(n,2), fn) .GT. -1)
91     fgroup = dens_index_a(nlabels(n,2), fn)
92     CALL getp(2, nlabels(n, 2), -1, fgroup, r(n), y(1))
93     density_thread(nlabels(n,3), fgroup) = density_thread(nlabels(n,3), fgroup) + y(1)
94     fn = fn + 1
95   END DO

```

```

96      END IF
97  END DO
98 !$OMP END DO
99 !$OMP CRITICAL
100 energy(1) = energy(1) + energy_thread(1)
101 density(:,:) = density(:,:) + density_thread(:,:)
102 forces(:,:) = forces(:,:) + forces_thread(:,:)
103 nl_force(:,:) = nl_force(:,:) + nl_force_thread(:,:)
104 !$OMP END CRITICAL
105 !$OMP END PARALLEL
106
107 ! Embedding Energy
108 ! If a small number of atoms, just use one thread
109 IF(cc .LE. 100)THEN
110   DO n = 1, cc
111     fn = 0
112     DO WHILE(embe_index_a(clabels(n), fn) .GT. -1)
113       fgroup = embe_index_a(clabels(n), fn)
114       CALL getp(3, clabels(n), -1, fgroup, density(n, fgroup), y(1))
115       energy(2) = energy(2) + y(1)
116       fn = fn + 1
117     END DO
118   END DO
119 ELSE
120 !$OMP PARALLEL &
121 !$OMP PRIVATE(n, y, fgroup, fn, energy_thread) &
122 !$OMP SHARED(cc, nlables, clabels, energy, density)
123   energy_thread(2) = 0.0D0
124 !$OMP DO
125   DO n = 1, cc
126     fn = 0
127     DO WHILE(embe_index_a(clabels(n), fn) .GT. -1)
128       fgroup = embe_index_a(clabels(n), fn)
129       CALL getp(3, clabels(n), -1, fgroup, density(n, fgroup), y(1))
130       energy_thread(2) = energy_thread(2) + y(1)
131       fn = fn + 1
132     END DO
133   END DO
134 !$OMP END DO
135 !$OMP CRITICAL
136   energy(2) = energy(2) + energy_thread(2)
137 !$OMP END CRITICAL
138 !$OMP END PARALLEL
139 END IF
140
141 ! Embedding Force
142 !$OMP PARALLEL &
143 !$OMP PRIVATE(n, fgroup, fn, forces_thread, dgradA, dgradB, fgradA, fgradB, fv) &
144 !$OMP SHARED(nc, nlables, rvec, density, forces)
145   forces_thread(:,:) = 0.0D0
146   nl_force_thread(:,:) = 0.0D0
147 !$OMP DO
148   DO n = 1, nc
149     dgradA(:) = 0.0D0
150     dgradB(:) = 0.0D0
151     fgradA(:) = 0.0D0
152     fgradB(:) = 0.0D0
153
154   ! Loop through groups and compute density and embedding gradients
155   fn = 0
156   DO WHILE(embe_index_a(nlabels(n, 1), fn) .GT. -1)

```

```

157   fgroup = embe_index_a(nlabels(n, 1), fn)
158   CALL getpgrad(2, nlabels(n,1), -1, fgroup, r(n), dgradA(fgroup))
159   CALL getpgrad(2, nlabels(n,2), -1, fgroup, r(n), dgradB(fgroup))
160   CALL getpgrad(3, nlabels(n,1), -1, fgroup, density(nlabels(n, 3), fgroup), fgradA(fgroup))
161   CALL getpgrad(3, nlabels(n,2), -1, fgroup, density(nlabels(n, 4), fgroup), fgradB(fgroup))
162   fn = fn + 1
163 END DO
164
165 ! Compute forces
166 fv(1:3) = SUM(fgradA(:) * dgradB(:) + fgradB(:) * dgradA(:)) * rvec(n, 1:3)
167 forces_thread(nlabels(n, 3), 1:3) = forces_thread(nlabels(n, 3), 1:3) + fv(1:3)
168 forces_thread(nlabels(n, 4), 1:3) = forces_thread(nlabels(n, 4), 1:3) - fv(1:3)
169 nl_force_thread(n, 1:3) = -fv(1:3)
170 END DO
171 !$OMP END DO
172 !$OMP CRITICAL
173 forces(:,:) = forces(:,:) + forces_thread(:,:)
174 nl_force(:,:) = nl_force(:,:) + nl_force_thread(:,:)
175 !$OMP END CRITICAL
176 !$OMP END PARALLEL
177
178 ! Energy (pair + embed)
179 energy(3) = SUM(energy(1:2))
180
181 ! Stress
182 DO n = 1, nc
183 IF(inhalo(n) .NE. 0)THEN
184   DO i = 1,3
185     DO j = 1,3
186       stress(i,j) = stress(i,j) + (rvec(n, i) * nl_force(n,j))
187     END DO
188   END DO
189 END IF
190 END DO
191 stress(1:3, 1:3) = stress(1:3, 1:3) / (2.0D0 * volume)
192
193
194
195 DEALLOCATE(density)
196 DEALLOCATE(density_thread)
197 DEALLOCATE(fgradA)
198 DEALLOCATE(fgradB)
199 DEALLOCATE(dgradA)
200 DEALLOCATE(dgradB)
201
202 END SUBROUTINE efs

```

S.2 Neighbour List Subroutine

Listing S.2: Energy, Force and Stress subroutine (Fortran)

```

1 SUBROUTINE nl(clabels, cin, a0, uv, rcut, nl_labels, r, rvec, nlhalo, nlsize)
2 !#####
3 IMPLICIT NONE
4 !#####
5 INTEGER(kind=StandardInteger), INTENT(IN) :: clabels(:)
6 REAL(kind=DoubleReal), INTENT(IN) :: cin(:,:)
7 REAL(kind=DoubleReal), INTENT(IN) :: a0
8 REAL(kind=DoubleReal), INTENT(IN) :: uv(1:3,1:3)

```

```

9  REAL(kind=DoubleReal), INTENT(IN) :: rcut
10 INTEGER(kind=StandardInteger), INTENT(INOUT) :: nl_labels(:,:)
11 REAL(kind=DoubleReal), INTENT(INOUT) :: r(:)
12 REAL(kind=DoubleReal), INTENT(INOUT) :: rvec(:, :)
13 INTEGER(kind=StandardInteger), INTENT(INOUT) :: nlhalo(:)
14 INTEGER(kind=StandardInteger), INTENT(OUT) :: nlsiz
15 !#####
16 REAL(kind=DoubleReal) :: ccoords(1:SIZE(cin,1),1:3)
17 REAL(kind=DoubleReal) :: gcoords(1:27*SIZE(cin,1),1:3)
18 INTEGER(kind=StandardInteger) :: glabels(1:27*SIZE(cin,1))
19 INTEGER(kind=StandardInteger) :: gids(1:27*SIZE(cin,1))
20 INTEGER(kind=StandardInteger) :: halo(1:27*SIZE(cin,1))
21 REAL(kind=DoubleReal) :: rcutsq
22 REAL(kind=DoubleReal) :: rsq
23 REAL(kind=DoubleReal) :: shift(1:3)
24 REAL(kind=DoubleReal) :: rvect(1:3)
25 INTEGER(kind=StandardInteger) :: m, n, i, j, k
26 !#####
27 DO n=1,SIZE(cin, 1)
28   ccoords(n, :) = a0 * MATMUL(uv, cin(n, :))
29 END DO
30
31 m = 0
32 DO i=-1,1
33   DO j=-1,1
34     DO k=-1,1
35       DO n=1, SIZE(cin, 1)
36         m = m + 1
37         shift(1) = i
38         shift(2) = j
39         shift(3) = k
40         gcoords(m, :) = a0 * MATMUL(uv, cin(n, :) + shift(:))
41         glabels(m) = clabels(n)
42         gids(m) = n
43         halo(m) = i + 3 * j + 9 * k
44     END DO
45   END DO
46 END DO
47
48 rcutsq = rcut * rcut
49
50 k = 0
51 DO n=1, SIZE(ccords, 1)
52   DO m=1, SIZE(gcoords, 1)
53     IF(gids(m) .GT. n)THEN
54       rvect(:) = gcoords(m, :) - ccoords(n, :)
55       rsq = SUM(rvect(:)**2)
56       IF(rsq .LE. rcutsq)THEN
57         k = k + 1
58         nl_labels(k, 1) = clabels(n)
59         nl_labels(k, 2) = glabels(m)
60         nl_labels(k, 3) = n
61         nl_labels(k, 4) = gids(m)
62         r(k) = sqrt(rsq)
63         rvec(k, :) = rvect(:) / r(k)
64         nlhalo(k) = halo(m)
65     END IF
66   END IF
67 END DO
68
69 END DO

```

```

70 nsize = k
71
72 !#####
73 END SUBROUTINE nl

```

S.3 Fortran Programmed Functions

Listing S.3: Polynomial Embedding (Fortran)

```

1  !# Polynomial embedding
2  !# General function for embedding functions
3  !# like F(r) = sqrt(r)
4  !#      F(r) = A sqrt(r) + B r**2 + C r**3
5
6
7  SUBROUTINE polynomial_embedding(r, p, p_fixed, y)
8  !#####
9  ! f(x) = (N r^3 e^{(-z r)})^2
10 IMPLICIT NONE
11 !#####
12 REAL(kind=DoubleReal), INTENT(IN) :: r
13 REAL(kind=DoubleReal), INTENT(IN) :: p(:)      !# Coeff
14 REAL(kind=DoubleReal), INTENT(IN) :: p_fixed(:) !# Power
15 REAL(kind=DoubleReal), INTENT(OUT) :: y
16 !#####
17 INTEGER(kind=StandardInteger) :: n
18 !#####
19 y = 0.0D0
20
21 IF(SIZE(p,1) .NE. SIZE(p_fixed,1))THEN
22   RETURN
23 END IF
24
25 DO n = 1, SIZE(p,1)
26   y = y + p(n) * r**p_fixed(n)
27 END DO
28
29 END SUBROUTINE polynomial_embedding
30
31
32
33 SUBROUTINE polynomial_embedding_v(r, p, p_fixed, y)
34 !#####
35 !
36 IMPLICIT NONE
37 !#####
38 REAL(kind=DoubleReal), INTENT(IN) :: r(:)
39 REAL(kind=DoubleReal), INTENT(IN) :: p(:)
40 REAL(kind=DoubleReal), INTENT(IN) :: p_fixed(:)
41 REAL(kind=DoubleReal), INTENT(OUT) :: y(1:SIZE(r,1))
42 INTEGER(kind=StandardInteger) :: n
43 !#####
44 DO n = 1, SIZE(r,1)
45   CALL polynomial_embedding(r(n), p, p_fixed, y(n))
46 END DO
47 END SUBROUTINE polynomial_embedding_v

```

Listing S.4: Polynomial Embedding Gradient (Fortran)

```

1  !# Polynomial embedding (GRAD)

```

```

2  !# General function for embedding functions
3  !# like F(r) = sqrt(r)
4  !#      F(r) = A sqrt(r) + B r**2 + C r**3
5
6
7  SUBROUTINE polynomial_embedding_grad(r, p, p_fixed, y)
8  !#####
9  ! f(x) = (N r^3 e^(-z r))^2
10 IMPLICIT NONE
11 !#####
12 REAL(kind=DoubleReal), INTENT(IN) :: r
13 REAL(kind=DoubleReal), INTENT(IN) :: p(:)      !# Coeff
14 REAL(kind=DoubleReal), INTENT(IN) :: p_fixed(:) !# Power
15 REAL(kind=DoubleReal), INTENT(OUT) :: y
16 !#####
17 INTEGER(kind=StandardInteger) :: n
18 !#####
19 y = 0.0D0
20
21 IF(SIZE(p,1) .NE. SIZE(p_fixed,1))THEN
22   RETURN
23 END IF
24
25 DO n = 1, SIZE(p,1)
26   y = y + (p(n) * p_fixed(n)) * r**(p_fixed(n) - 1.0D0)
27 END DO
28
29 END SUBROUTINE polynomial_embedding_grad
30
31
32
33 SUBROUTINE polynomial_embedding_grad_v(r, p, p_fixed, y)
34 !#####
35 !
36 IMPLICIT NONE
37 !#####
38 REAL(kind=DoubleReal), INTENT(IN) :: r(:)
39 REAL(kind=DoubleReal), INTENT(IN) :: p(:)
40 REAL(kind=DoubleReal), INTENT(IN) :: p_fixed(:)
41 REAL(kind=DoubleReal), INTENT(OUT) :: y(1:SIZE(r,1))
42 INTEGER(kind=StandardInteger) :: n
43 !#####
44 DO n = 1, SIZE(r,1)
45   CALL polynomial_embedding_grad(r(n), p, p_fixed, y(n))
46 END DO
47 END SUBROUTINE polynomial_embedding_grad_v

```

Listing S.5: Spline (Fortran)

```

1 ! POWER SPLINE
2 ! sum (ai (r - ri)^3 H(ri - r)) - cubic example
3 ! sum (ai (r - ri)^5 H(ri - r)) - quintic example
4
5 ! SCALAR SUBROUTINE
6 SUBROUTINE spline(r, p, p_fixed, y)
7 !#####
8 ! p coefficients
9 ! pf r cutoffs
10 ! they must be the same size
11 IMPLICIT NONE
12 !#####
13 REAL(kind=DoubleReal), INTENT(IN) :: r

```

```

14  REAL(kind=DoubleReal), INTENT(IN) :: p(:)
15  REAL(kind=DoubleReal), INTENT(IN) :: p_fixed(:)
16  REAL(kind=DoubleReal), INTENT(OUT) :: y
17  !#####
18  INTEGER(kind=StandardInteger) :: ps, pfs, ss
19  REAL(kind=DoubleReal) :: settings(1:10)
20  REAL(kind=DoubleReal) :: nodes(1:SIZE(p,1))
21  REAL(kind=DoubleReal) :: rk
22  INTEGER(kind=StandardInteger) :: n, k
23  ! Settings
24  INTEGER(kind=StandardInteger) :: power_lower
25  INTEGER(kind=StandardInteger) :: power_upper
26  INTEGER(kind=StandardInteger) :: rkoption
27  REAL(kind=DoubleReal) :: zbl_qa
28  REAL(kind=DoubleReal) :: zbl_qb
29  REAL(kind=DoubleReal) :: h_node_r = 0.0D0
30  REAL(kind=DoubleReal) :: h_node_c = 0.0D0
31  INTEGER(kind=StandardInteger) :: h_node_p
32  REAL(kind=DoubleReal) :: yzbl
33  !#####
34  y = 0.0D0
35  ss = 10
36  ps = SIZE(p,1)
37  pfs = SIZE(p_fixed,1)
38
39  IF(pfs .LT. ss)THEN
40    RETURN
41  END IF
42
43  !Read Settings
44  settings(1:ss) = p_fixed(1:ss)
45  power_lower = INT(settings(1)) ! 3, 5 etc
46  power_upper = INT(settings(2)) !
47  rkoption = INT(settings(3)) ! 0 = (rk - r)^n 1 = (r - rk)^n
48  zbl_qa = settings(4)
49  zbl_qb = settings(5)
50  h_node_r = settings(6) ! Hardening node rcutoff
51  h_node_c = settings(7) ! Hardening node coefficient
52  h_node_p = INT(settings(8)) ! Hardening node power
53
54
55  IF(pfs .NE. (ps + ss))THEN
56    RETURN
57  END IF
58  nodes(1:ps) = p_fixed(ss+1:pfs)
59
60  IF(r .GT. nodes(ps))THEN
61    RETURN
62  END IF
63
64  ! Hardening node (See G. Bonny 2012 code)
65  IF(h_node_r .GT. 0.0D0 .AND. r .LE. h_node_r)Then
66    y = y + h_node_c * (h_node_r - r)**h_node_p
67  END IF
68
69  ! Power spline
70  k = power_lower
71  DO n = 1, ps
72    rk = nodes(n)
73    IF((rk - r) .GE. 0.0D0)THEN
74      IF(rkoption .EQ. 0)THEN

```

```

75      y = y + p(n) * (rk - r)**k
76      ELSE
77          y = y + p(n) * (r - rk)**k
78      END IF
79  END IF
80  IF(k .LT. power_upper)THEN
81      k = k + 1
82  ELSE
83      k = power_lower
84  END IF
85 END DO
86
87 ! Add ZBL
88 IF(zbl_qa .GT. 0.0D0 .AND. zbl_qb .GT. 0.0D0)THEN
89     CALL fzbl(r, zbl_qa, zbl_qb, yzbl)
90     y = y + yzbl
91 END IF
92
93
94
95
96
97
98 END SUBROUTINE spline
99
100
101 ! VECTOR SUBROUTINE
102 SUBROUTINE spline_v(r, p, p_fixed, y)
103 !#####
104 ! CUBIC SPLINE
105 IMPLICIT NONE
106 !#####
107 REAL(kind=DoubleReal), INTENT(IN) :: r(:)
108 REAL(kind=DoubleReal), INTENT(IN) :: p(:)
109 REAL(kind=DoubleReal), INTENT(IN) :: p_fixed(:)
110 REAL(kind=DoubleReal), INTENT(OUT) :: y(1:SIZE(r,1))
111 INTEGER(kind=StandardInteger) :: n
112 !#####
113 ! Loop through all the values in r(:), calculate and store in y(:)
114 DO n = 1, SIZE(r,1)
115     CALL spline(r(n), p, p_fixed, y(n))
116 END DO
117
118 END SUBROUTINE spline_v

```

S.4 Main fitting class

Listing S.6: Fitting Class (Python)

```

1 #!/bin/python3
2
3 import numpy
4 import time
5 import os
6 import sys
7
8 from scipy.optimize import minimize
9 from scipy.optimize import basinhopping
10

```

```

11  from f_toolbox import atom
12  from f_toolbox import math
13  from f_toolbox import transforms
14
15  from g import g
16  from ds import ds
17  from display import display
18  from configs import configs
19  from nl import nl
20  from output import output
21  from plot import plot
22  from relax import relax
23  from eos import eos
24  from bp import bp
25  from units import units
26  from potential import potential
27  from rss import rss
28  from search_step import search_step
29  from simulated_annealing import simulated_annealing
30  from genetic_algorithm import ga
31  from random_search import random_search
32  from hybrid_search import hybrid_search
33  from wide_search import wide_search
34
35  class potfit:
36
37
38      @staticmethod
39      def run():
40          # Log
41          output.log("Potfit", verbose=0)
42
43          # Make dirs
44          potfit.make_dirs()
45
46          # Turn off terminal log
47          g.verbose['print'] = -1
48
49          # Create storage dictionary
50          potfit.set_potfit_store()
51
52          # Parameters
53          p = potential.get_p()
54
55
56          if(p is None):
57              output.log("No parameters to optimise", verbose=0)
58              output.log("End", verbose=0)
59              exit()
60
61          g.potfit['potfit_start_time'] = time.time()
62          for sn in range(len(g.potfit_steps)):
63              if(sn > 0):
64                  p = ds.potential_p(g.potfit['p_best'])
65                  g.potfit['sn'] = sn
66                  g.potfit['sn_counter'] = 0
67
68
69          g.potfit_steps[sn]['stats_rss'][0] = g.potfit['rss_best']
70          g.potfit_steps[sn]['stats_start_time'] = time.time()
71          g.potfit_steps[sn]['stats_start_counter'] = g.calc_counter

```

```

72
73
74     # Run
75     if(g.potfit_steps[sn]['type'].lower() == 'rand'):
76         p = random_search.run(potfit.get_rss, p, niter=g.potfit_steps[sn]['niter'])
77     elif(g.potfit_steps[sn]['type'].lower() == 'ws'):
78         p = wide_search.run(potfit.get_rss, p)
79     elif(g.potfit_steps[sn]['type'].lower() == 'sa'):
80         p = simulated_annealing.run(potfit.get_rss, p,
81                                     niter=g.potfit_steps[sn]['niter'], titer=g.potfit_steps[sn]['titer'], tstart=g.
82                                     potfit_steps[sn]['tstart'], tend=g.potfit_steps[sn]['tend'], pfact=g.
83                                     potfit_steps[sn]['pfact'], pvar=g.potfit_steps[sn]['pvar'], vartype=g.
84                                     potfit_steps[sn]['vartype'], gaussian=g.potfit_steps[sn]['gaussian'])
85     elif(g.potfit_steps[sn]['type'].lower() == 'ga'):
86         p = ga.run(potfit.get_rss, p, niter=g.potfit_steps[sn]['niter'], popsize=g.potfit_steps[sn]['popsize'],
87                     fresh=g.potfit_steps[sn]['fresh'], threshold=1.0e20, min_final=True)
88     elif(g.potfit_steps[sn]['type'].lower() == 'hs'):
89         p = hybrid_search.run(potfit.get_rss, p,
90                               pool_size=g.potfit_steps[sn]['poolsize'],
91                               sample_size=g.potfit_steps[sn]['samplesize'],
92                               minima_size=g.potfit_steps[sn]['minsize'])
93     elif(g.potfit_steps[sn]['type'].lower() == 'bh'):
94         res = basinhopping(potfit.get_rss, p, niter=g.potfit_steps[sn]['niter'], T=10.0, stepsize=1.0)
95         p = res['x']
96     elif(g.potfit_steps[sn]['type'].lower() == 'bfgs'):
97         res = minimize(potfit.get_rss, p, method='BFGS', options={'gtol': 1.0e-6, 'maxiter': 4, })
98         p = res['x']
99     elif(g.potfit_steps[sn]['type'].lower() == 'cg'):
100        res = minimize(potfit.get_rss, p, method='cg', options={'gtol': 1.0e-6, 'maxiter': 4, })
101        p = res['x']
102
103     # Record stats
104     g.potfit_steps[sn]['stats_rss'][1] = g.potfit['rss_best']
105     g.potfit_steps[sn]['stats_time'] = time.time() - g.potfit_steps[sn]['stats_start_time']
106     g.potfit_steps[sn]['stats_counter'] = g.calc_counter - g.potfit_steps[sn]['stats_start_counter']
107     g.potfit_steps[sn]['stats_speed'] = g.potfit_steps[sn]['stats_counter'] / g.potfit_steps[sn]['stats_time']
108     g.potfit_steps[sn]['stats_complete'] = True
109
110
111
112     @staticmethod
113     def end(interupt=False):
114         g.potfit['potfit_end_time'] = time.time()
115
116         if(interupt):
117             sn = g.potfit['sn']
118             g.potfit_steps[sn]['stats_rss'][1] = g.potfit['rss_best']
119             g.potfit_steps[sn]['stats_time'] = time.time() - g.potfit_steps[sn]['stats_start_time']
120             g.potfit_steps[sn]['stats_counter'] = g.calc_counter - g.potfit_steps[sn]['stats_start_counter']
121             g.potfit_steps[sn]['stats_speed'] = g.potfit_steps[sn]['stats_counter'] / g.potfit_steps[sn]['stats_time']
122             g.potfit_steps[sn]['stats_complete'] = False
123
124
125         output.log("#####", verbose=0)
126         output.log("Ending of Potential Fit", verbose=0)
127         output.log("#####", verbose=0)
128         output.log("", verbose=0)

```

```

129     if(interrupt):
130         output.log("Reached maximum time limit", verbose=0)
131     else:
132         output.log("Completed within maximum time limit", verbose=0)
133     output.log("", verbose=0)
134
135     # Set parameters for potential
136     p = ds.potential_p(g.potfit['p_best'])
137     potential.update_p(p)
138
139     # Run calculation
140     efs_rss, bp_rss, total_rss = potfit.get_rss_calc()
141     output.log("", verbose=0)
142
143
144     output.log("Best Parameters", verbose=0)
145     output.log("-----", verbose=0)
146     line = ''
147     for i in range(len(g.potfit['p_best'])):
148         line = line + "{:10.3e} ".format(g.potfit['p_best'][i])
149         if((i + 1) % 10 == 0 and i != len(g.potfit['p_best'])-1):
150             output.log(line, verbose=0)
151             line = ''
152     if(line != ''):
153         output.log(line, verbose=0)
154
155     t = g.potfit['potfit_end_time'] - g.potfit['potfit_start_time']
156     output.log("Stats", verbose=0)
157     output.log("-----", verbose=0)
158     output.log("{:12s} {:16.2f}".format("Time: ", t), verbose=0)
159     output.log("{:12s} {:16d} {:16.2f}".format("Steps: ", g.potfit['counter'], g.potfit['counter']/ t), verbose=0)
160     output.log("{:12s} {:16d} {:16.2f}".format("Configs: ", g.calc_counter, g.calc_counter/ t), verbose=0)
161     output.log("{:12s} {:16.2f}".format("Best RSS: ", g.potfit['rss_best']), verbose=0)
162     output.log('', verbose=0)
163
164     # Make tabulated
165     potential.tab_for_output()
166
167     # Output potentials - analytic, tab, and plots
168     output.potentials(g.dir['fit_best_pot'])
169     output.potentials_tabulated(g.dir['fit_best_pottab'])
170     plot.potentials(g.dir['fit_best_plot'])
171
172     # Output bp and other plots
173     output.bp(g.dir['fit_best'])
174     plot.plot_eos(g.dir['fit_best_plots'])
175     plot.plot_ec(g.dir['fit_best_plots'])
176     plot.plot_rose(g.dir['fit_best_plots'])
177     plot.plot_msp(g.dir['fit_best_plots'])
178
179     output.potfit_summary(g.dir['fit'])
180
181
182     # Turn on terminal log
183     g.verbose['print'] = 1
184     exit()
185
186     sys.exit()
187
188
189

```

```

190     @staticmethod
191     def make_dirs():
192
193         g.dir['fit_best'] = os.path.join(g.dir['fit'], "best")
194         g.dir['fit_best_pot'] = os.path.join(g.dir['fit_best'], "pot")
195         g.dir['fit_best_pottab'] = os.path.join(g.dir['fit_best'], "pot_tab")
196         g.dir['fit_best_plot'] = os.path.join(g.dir['fit_best'], "pot_plot")
197         g.dir['fit_best_plots'] = os.path.join(g.dir['fit_best'], "plots")
198
199         g.dir['fit_temp_best'] = os.path.join(g.dir['fit'], "temp")
200
201     for k in g.dir.keys():
202         os.makedirs(g.dir[k], exist_ok=True)
203
204
205
206     @staticmethod
207     def temporary_save():
208         tp = g.potfit_bestsavoperiod
209         dirroot = g.dir['fit_temp_best']
210         tn = str(int(tp * numpy.ceil((time.time() - g.start) / tp)))
211         while(len(tn) < 8):
212             tn = "0" + tn
213
214         tndir = os.path.join(dirroot, tn)
215         tndir_pot = os.path.join(tndir, "pot")
216         tndir_pottab = os.path.join(tndir, "pot_tab")
217         tndir_plot = os.path.join(tndir, "pot_plot")
218         os.makedirs(tndir, exist_ok=True)
219         os.makedirs(tndir_pot, exist_ok=True)
220         os.makedirs(tndir_pottab, exist_ok=True)
221         os.makedirs(tndir_plot, exist_ok=True)
222
223         output.potentials(tndir_pot)
224         potential.tab_for_output()
225         output.potentials_tabulated(tndir_pottab)
226         plot.potentials(tndir_plot)
227         output.bp(tndir_pot)
228
229
230     # Function to minimise
231     @staticmethod
232     def get_rss(p_in):
233         # End
234         if((time.time() - g.start) > g.maxtime):
235             potfit.end(interupt=True)
236
237         # Set parameters for potential
238         p = ds.potential_p(p_in)
239         potential.update_p(p)
240
241         # Run calculation
242         efs_rss, bp_rss, total_rss = potfit.get_rss_calc()
243
244         # Save results
245         g.potfit['counter'] = g.potfit['counter'] + 1
246         g.potfit['improvement_counter'] = g.potfit['improvement_counter'] + 1
247         g.potfit['sn_counter'] = g.potfit['sn_counter'] + 1
248         g.potfit_steps[g.potfit['sn']] ['counter'] = g.potfit_steps[g.potfit['sn']] ['counter'] + 1
249         g.potfit['p_current'] = numpy.copy(p)
250         g.potfit['efs_rss_current'] = efs_rss

```

```

251     g.potfit['bp_rss_current'] = bp_rss
252     g.potfit['rss_current'] = total_rss
253
254     for n in range(len(g.bp)):
255         g.potfit['bp_current'][n]['a0'] = g.bp[n]['c_a0']
256         g.potfit['bp_current'][n]['e0'] = g.bp[n]['c_e0']
257         g.potfit['bp_current'][n]['v0'] = g.bp[n]['c_v0']
258         g.potfit['bp_current'][n]['b0_gpa'] = g.bp[n]['c_b0_gpa']
259         g.potfit['bp_current'][n]['ec_gpa'] = numpy.copy(g.bp[n]['c_ec_gpa'])
260         g.potfit['bp_current'][n]['mspec_gpa'] = numpy.copy(g.bp[n]['c_msp_ec_gpa'])
261         g.potfit['bp_current'][n]['rss_details'] = g.bp[n]['rss_details']
262
263     # Save best
264     if((g.potfit['rss_best'] is None or total_rss < g.potfit['rss_best']) and not numpy.isnan(total_rss)):
265         g.potfit['improvement_counter'] = 0
266         g.potfit['p_best'] = numpy.copy(p)
267         g.potfit['efs_rss_best'] = efs_rss
268         g.potfit['bp_rss_best'] = bp_rss
269         g.potfit['rss_best'] = total_rss
270
271     # Save best values
272     for n in range(len(g.bp)):
273         g.potfit['bp_best'][n]['a0'] = g.bp[n]['c_a0']
274         g.potfit['bp_best'][n]['e0'] = g.bp[n]['c_e0']
275         g.potfit['bp_best'][n]['v0'] = g.bp[n]['c_v0']
276         g.potfit['bp_best'][n]['b0_gpa'] = g.bp[n]['c_b0_gpa']
277         g.potfit['bp_best'][n]['ec_gpa'] = numpy.copy(g.bp[n]['c_ec_gpa'])
278         g.potfit['bp_best'][n]['mspec_gpa'] = numpy.copy(g.bp[n]['c_msp_ec_gpa'])
279
280     potfit.temporary_save()
281
282
283
284     # Display results
285     display.show()
286
287     # Log
288     output.log("Total rss: " + str(total_rss) + " " + str(g.potfit['rss_best']), verbose=2)
289     output.log("Total rss: " + str(p), verbose=2)
290
291     #
292     return total_rss
293
294
295
296 @staticmethod
297 def get_rss_calc():
298     efs_rss = 0.0
299     if(configs.count('efs') > 0 and g.rss_w['configs'] != 0.0):
300         configs.calc('efs')
301         efs_rss = configs.rss(config_tag='efs')
302     bp_rss = 0.0
303     if(len(g.bp) > 0 and g.rss_w['bp'] != 0.0):
304         bp_rss = bp.run_rss()
305     total_rss = bp_rss + efs_rss
306
307
308     return efs_rss, bp_rss, total_rss
309
310
311

```

```

312     @staticmethod
313     def set_potfit_store():
314         g.potfit = ds.potfit()
315
316     # BP current
317     if(g.potfit['bp_current'] is None):
318         g.potfit['bp_current'] = []
319         for n in range(len(g.bp)):
320             g.potfit['bp_current'].append(ds.potfit_bp())
321
322     # BP best
323     if(g.potfit['bp_best'] is None):
324         g.potfit['bp_best'] = []
325         for n in range(len(g.bp)):
326             g.potfit['bp_best'].append(ds.potfit_bp())

```

S.5 Global Fitting Algorithms

Listing S.7: Simulated Annealing Class (Python)

```

1  #!/bin/python3
2
3  import numpy
4  import time
5  import copy
6
7  from g import g
8  from ds import ds
9
10 class simulated_annealing:
11
12
13     @staticmethod
14     def run(f, p, niter=1000, titer=10, tstart=100, tend=0.01, pfact=0.7, pvar=None, vartype=None, gaussian=False):
15
16         if(pvar == None or vartype == None):
17             g.displaynote = 'Initializing pvar'
18             p, pvar = simulated_annealing.pvar(f, p)
19         elif(vartype == 'add'):
20             pvar = pvar + numpy.zeros((len(p),), dtype=numpy.float64)
21         elif(vartype == 'mult'):
22             pvar = pvar * p
23         p = simulated_annealing.optimize(f, p, pvar, niter, titer, tstart, tend, pfact, gaussian)
24         g.displaynote = ''
25
26     return p
26
27     @staticmethod
28     def optimize(f, p, pvar, niter=1000, titer=10, tstart=100, tend=0.01, pfact=0.7, gaussian=False):
29
30         temp = numpy.linspace(tstart, tend, titer)
31
32         rss = f(p)
33         rss_best = rss
34         p_best = numpy.copy(p)
35
36         for tn in range(titer):
37             t = temp[tn]
38             dp = pfact**tn * 10.0 * pvar[:]

```

```

40     for n in range(niter):
41         g.displaynote = 'T Loop ' + str(tn) + " N Loop " + str(n) + " t=" + str("{0:7.3f}".format(t)) + " pfact=" +
42             str("{0:7.3f}").format(pfact**(tn)))
43         p_new = numpy.copy(p)
44
45         if(gaussian):
46             p_new[:] = p_new[:] + dp * numpy.random.normal(0.0, 0.1, len(p_new[:]))
47         else:
48             p_new[:] = p_new[:] + dp * (0.5 - numpy.random.rand(len(p_new[:])))
49
50         rss_new = f(p_new)
51         if(rss_new < rss or numpy.random.uniform() < numpy.exp((rss_new - rss) / t)):
52             rss = rss_new
53             p = numpy.copy(p_new)
54             if(rss < rss_best):
55                 rss_best = rss_new
56                 p_best = numpy.copy(p_new)
57             p = numpy.copy(p_best)
58             rss = rss_best
59         return p_best
60
61
62 @staticmethod
63 def pvar(f, p):
64     p_upper = numpy.zeros((len(p),), dtype=numpy.float64)
65     p_lower = numpy.zeros((len(p),), dtype=numpy.float64)
66
67     rss_initial = f(p)
68     var = numpy.zeros((14,), dtype=numpy.float64)
69     for n in range(14):
70         var[n] = 10**(-7 + n)
71
72     for pn in range(len(p)):
73         for n in range(14):
74             p_new = numpy.copy(p)
75             p_new[pn] = p_new[pn] + var[n]
76             rss = f(p_new)
77             if(rss > 10.0 * rss_initial):
78                 p_upper[pn] = p_new[pn]
79                 break
80     for pn in range(len(p)):
81         for n in range(14):
82             p_new = numpy.copy(p)
83             p_new[pn] = p_new[pn] - var[n]
84             rss = f(p_new)
85             if(rss > 10.0 * rss_initial):
86                 p_lower[pn] = p_new[pn]
87                 break
88
89
90     pvar = p_upper - p_lower
91     p = p_lower + 0.5 * pvar
92     return p, pvar

```

Listing S.8: Genetic Algorithm Class (Python)

```

1 #!/bin/python3
2
3 import numpy
4 import time

```

```

5 import copy
6 from scipy.optimize import minimize
7
8 from g import g
9
10 class ga:
11
12     f = None
13     p_in = None
14     p_best = None
15     rss_best = None
16     psize = 0
17     popsize = 0
18     c_popsizes = 0
19     pop = None
20     pop_rss = None
21     f_pop = None
22     f_pop_rss = None
23     f_popsizes = 0
24     c_pop = None
25     c_pop_rss = None
26     threshold = 1.0e20
27     parents = None
28     populate_counter = 0
29     evolve_period = 10
30     evolve_amount = 3
31     search = None
32
33 @staticmethod
34 def run(f, p, niter=1000, popsize=1000, fresh=0.1, threshold=1.0e20, min_final=True, search=None):
35     ga.f = f
36     ga.p_in = numpy.copy(p)
37     ga.psize = len(p)
38     ga.popsize = popsize
39     ga.c_popsizes = popsize
40     ga.f_popsizes = int(popsize * fresh)
41     if(search is None):
42         ga.search = 'w'
43     else:
44         ga.search = search
45
46     ga.setup()
47     ga.pop, ga.pop_rss = ga.populate(ga.pop, ga.pop_rss)
48
49     for gn in range(niter):
50         ga.generation(gn)
51         p = numpy.copy(ga.pop[0, :])
52         if(min_final):
53             res = minimize(ga.rss, p, method='BFGS', options={'gtol': 1.0e-8, 'maxiter': 10, })
54             p = res['x']
55
56         # Clear note
57         g.displaynote = ''
58         return p
59
60 @staticmethod
61 def rss(p):
62     rss = ga.f(p)
63     if(ga.rss_best is None):
64         ga.p_best = numpy.copy(p)
65         ga.rss_best = rss

```

```

66     if(rss < ga.rss_best):
67         ga.p_best = numpy.copy(p)
68         ga.rss_best = rss
69     return rss
70
71 @staticmethod
72 def generation(gn):
73     # Shuffle parent list
74     ga.parents = numpy.arange(ga.popsize)
75     numpy.random.shuffle(ga.parents)
76
77     # Breed
78
79     g.displaynote = 'gen ' + str(gn) + ' breed parents'
80     for n in range(ga.popsize // 2):
81         pa = numpy.copy(ga.pop[ga.parents[2 * n], :])
82         pb = numpy.copy(ga.pop[ga.parents[2 * n + 1], :])
83         ca, cb = ga.breed(pa, pb)
84         ca = ga.no_clones(ca)
85         cb = ga.no_clones(cb)
86         ga.c_pop[2 * n, :] = numpy.copy(ca[:])
87         ga.c_pop[2 * n+1, :] = numpy.copy(cb[:])
88         ga.c_pop_rss[2 * n] = ga.rss(ca)
89         ga.c_pop_rss[2 * n+1] = ga.rss(cb)
90
91     # Merge
92     ga.merge(ga.pop, ga.pop_rss, ga.c_pop, ga.c_pop_rss)
93
94
95     # Shuffle parent list
96     ga.parents = numpy.arange(ga.popsize)
97     numpy.random.shuffle(ga.parents)
98
99     # Fresh
100    g.displaynote = 'gen ' + str(gn) + ' populate fresh'
101    ga.f_pop, ga.f_pop_rss = ga.populate(ga.f_pop, ga.f_pop_rss)
102
103    g.displaynote = 'gen ' + str(gn) + ' breed fresh'
104    # Breed
105    for n in range(ga.f_popsize):
106        loop = True
107        while(loop):
108            try:
109                pa = numpy.copy(ga.pop[ga.parents[n], :])
110                pb = numpy.copy(ga.f_pop[n, :])
111                ca, cb = ga.breed(pa, pb)
112                ca = ga.no_clones(ca)
113                cb = ga.no_clones(cb)
114                ga.c_pop[2 * n, :] = numpy.copy(ca[:])
115                ga.c_pop[2 * n+1, :] = numpy.copy(cb[:])
116                ga.c_pop_rss[2 * n] = ga.rss(ca)
117                ga.c_pop_rss[2 * n+1] = ga.rss(cb)
118                loop = False
119            except:
120                pass
121
122    # Merge
123    ga.merge(ga.pop, ga.pop_rss, ga.c_pop[:2*ga.f_popsize, :], ga.c_pop_rss[:2 * ga.f_popsize])
124
125    if(ga.evolve_period is not None):
126        if(gn > 0 and (gn+1)%ga.evolve_period == 0):

```

```

127
128     # Shuffle parent list
129     ga.parents = numpy.arange(ga.popsize)
130     numpy.random.shuffle(ga.parents[2:])
131
132     for en in range(ga.evolve_amount):
133         g.displaynote = 'gen ' + str(gn) + ' evolve ' + str(en)
134         pa = numpy.copy(ga.pop[ga.parents[en], :])
135         res = minimize(ga.rss, pa, method='CG', options={'gtol': 1.0e-4, 'maxiter': 1, })
136         pe = res['x']
137         ga.e_pop[en, :] = pe[:]
138         ga.e_pop_rss[en] = ga.rss(pe[:])
139
140     # Merge
141     ga.merge(ga.pop, ga.pop_rss, ga.e_pop[:, :], ga.e_pop_rss[:])
142
143 @staticmethod
144 def breed(pa, pb):
145     ca = numpy.zeros((ga.psize,), dtype=numpy.float64,)
146     cb = numpy.zeros((ga.psize,), dtype=numpy.float64,)
147     state = 0
148     for n in range(ga.psize):
149         if(state == 0):
150             ca[n] = pa[n]
151             cb[n] = pb[n]
152         else:
153             cb[n] = pa[n]
154             ca[n] = pb[n]
155         if(numpy.random.uniform() > 0.5):
156             if(state == 0):
157                 state = 1
158             else:
159                 state = 0
160     return ca, cb
161
162
163 @staticmethod
164 def no_clones(p):
165     for n in range(ga.popsize):
166         if(p[:,].all() == ga.pop[n, :].all()):
167             p[:] = numpy.random.uniform() * p[:]
168     return p[:]
169
170
171 @staticmethod
172 def setup():
173     ga.pop = numpy.zeros((ga.popsize, ga.psize,), dtype=numpy.float64,)
174     ga.pop_rss = numpy.zeros((ga.popsize,), dtype=numpy.float64,)
175     ga.c_pop = numpy.zeros((ga.popsize, ga.psize,), dtype=numpy.float64,)
176     ga.c_pop_rss = numpy.zeros((ga.popsize,), dtype=numpy.float64,)
177     ga.f_pop = numpy.zeros((ga.f_popsize, ga.psize,), dtype=numpy.float64,)
178     ga.f_pop_rss = numpy.zeros((ga.f_popsize,), dtype=numpy.float64,)
179     ga.e_pop = numpy.zeros((ga.evolve_amount, ga.psize,), dtype=numpy.float64,)
180     ga.e_pop_rss = numpy.zeros((ga.evolve_amount,), dtype=numpy.float64,)
181
182
183 @staticmethod
184 def populate(pop, pop_rss):
185     g.displaynote = 'populating (' + str(len(pop)) + ')'
186     ga.populate_counter = ga.populate_counter + 1
187     ns = 0

```

```

188     if(ga.populate_counter == 1):
189         ns = 1
190         pop[0,:] = ga.p_in[:]
191         pop_rss[0] = ga.rss(pop[0,:])
192         for n in range(ns, len(pop_rss)):
193             loop = True
194             while(loop):
195                 r = numpy.random.uniform()
196                 # n narrow search
197                 if(ga.search.lower() == 'n'):
198                     if(r < 0.05):
199                         p_new = ga.p_in + 10**(-4*numpy.random.uniform()) * ga.p_in * (0.5 - numpy.random.rand(ga.psize))
200                     elif(r < 0.25):
201                         p_new = ga.p_best + 0.01 * ga.p_best * (0.5 - numpy.random.rand(ga.psize))
202                     elif(r < 0.75):
203                         p_new = ga.p_best + 10**(-1-4*numpy.random.uniform()) * ga.p_best * (0.5 - numpy.random.rand(ga.psize))
204                     elif(r < 0.90):
205                         p_new = ga.p_best + 10**(-1-6*numpy.random.uniform()) * ga.p_best * (0.5 - numpy.random.rand(ga.psize))
206                     # 10% from random
207                     else:
208                         p_new = (0.5 - numpy.random.rand(ga.psize)) * 10**(7.0 * (0.5 - numpy.random.rand(ga.psize)))
209                 # w wider search
210                 else:
211                     if(r < 0.05):
212                         p_new = ga.p_in + 10**(-4*numpy.random.uniform()) * ga.p_in * (0.5 - numpy.random.rand(ga.psize))
213                     elif(r < 0.12):
214                         p_new = ga.p_best + 0.01 * ga.p_best * (0.5 - numpy.random.rand(ga.psize))
215                     elif(r < 0.24):
216                         p_new = ga.p_best + 10**(-1-4*numpy.random.uniform()) * ga.p_best * (0.5 - numpy.random.rand(ga.psize))
217                     elif(r < 0.3):
218                         p_new = ga.p_best + 10**(-1-6*numpy.random.uniform()) * ga.p_best * (0.5 - numpy.random.rand(ga.psize))
219                     # 10% small parameters
220                     elif(r < 0.4):
221                         p_new = 1.0e-5 * (0.5 - numpy.random.rand(ga.psize))
222                     # 10% large parameters
223                     elif(r < 0.5):
224                         p_new = 1.0e3 * (0.5 - numpy.random.rand(ga.psize))
225                     # 25% random
226                     elif(r < 0.75):
227                         p_new = (0.5 - numpy.random.rand(ga.psize)) * 10**(5.0 * (0.5 - numpy.random.uniform()))
228                     # 25% random
229                     elif(r <= 1.0):
230                         p_new = (0.5 - numpy.random.rand(ga.psize)) * 10**(7.0 * (0.5 - numpy.random.rand(ga.psize)))
231             rss = ga.rss(p_new)
232
233             if(rss < ga.threshold or numpy.isnan(rss)):
234                 loop = False
235                 pop[n,:] = p_new[:]
236                 pop_rss[n] = rss
237
238     return pop, pop_rss
239
240
241 @staticmethod
242 def merge(pop_a, rss_a, pop_b, rss_b):
243     ax = pop_a.shape[0]
244     ay = pop_a.shape[1]
245     bx = pop_b.shape[0]
246     by = pop_b.shape[1]
247
248     t = numpy.zeros((ax + bx, by + 1,), dtype=numpy.float64,)
```

```

249     t[:ax,0] = rss_a[:,]
250     t[ax:,0] = rss_b[:,]
251     t[:ax,1:] = pop_a[:, :]
252     t[ax:,1:] = pop_b[:, :]

253
254     t = t[t[:, 0].argsort()]
255     rss_a[:, :] = t[:ax,0]
256     pop_a[:, :] = t[:ax, 1:]

257
258
259     rss_b[:, :] = 0.0
260     pop_b[:, :] = 0.0

```

S.6 Bulk Properties

Listing S.9: Bulk Properties Class (Python)

```

1  #!/bin/python3
2
3  import numpy
4  import time
5  import os
6
7  from f_toolbox import atom
8  from f_toolbox import math
9  from f_toolbox import transforms
10
11 from g import g
12 from ds import ds
13 from configs import configs
14 from potential import potential
15 from nl import nl
16 from output import output
17 from plot import plot
18 from relax import relax
19 from eos import eos
20 from units import units
21 from rose import rose
22 from msp import msp
23
24 class bp:
25
26     log = []
27
28
29     @staticmethod
30     def run():
31         output.log("Run BP Calculation", verbose=0)
32
33         bp.make_dirs()
34         bp.build()
35         bp.calc()
36         bp.rss()
37         bp.output()
38         output.log("End", verbose=0)
39         exit()
40
41
42

```

```

43     @staticmethod
44     def run_rss():
45         bp.build()
46         bp.calc()
47         return bp.rss()
48
49
50     @staticmethod
51     def output():
52         bp.make_dirs()
53
54         # Output summary
55         d = output.bp(g.dir['bp'])
56         output.log(d, verbose=0)
57
58         # EoS, EC, Rose, MSP plots
59         plot.plot_eos(g.dir['bp_plots'])
60         plot.plot_ec(g.dir['bp_plots'])
61         plot.plot_rose(g.dir['bp_plots'])
62         plot.plot_msp(g.dir['bp_plots'])
63
64         # Potential data and plots
65         potential.tab_for_output()
66         output.potentials(g.dir['bp_pot'])
67         output.potentials_tabulated(g.dir['bp_pot_data'])
68         plot.potentials(g.dir['bp_pot_plots'])
69
70
71     # Make additional directories for bp
72     @staticmethod
73     def make_dirs():
74
75         g.dir['bp_plots'] = os.path.join(g.dir['bp'], "plots")
76         g.dir['bp_pot'] = os.path.join(g.dir['bp'], "pot")
77         g.dir['bp_pot_plots'] = os.path.join(g.dir['bp'], "pot_plots")
78         g.dir['bp_pot_data'] = os.path.join(g.dir['bp'], "pot_data")
79
80         for k in g.dir.keys():
81             os.makedirs(g.dir[k], exist_ok=True)
82
83
84     """
85     Calculate RSS
86     #####
87
88     @staticmethod
89     def rss():
90         total_rss = 0.0
91         output.log("Calculate BP RSS", verbose=3)
92         for n in range(len(g.bp)):
93             rss_v = bp.rss_inner(n)
94             output.log("BP " + str(n) + " " + str(rss_v), verbose=2)
95             total_rss = total_rss + rss_v
96         output.log("BP total rss " + str(total_rss), verbose=2)
97         return total_rss
98
99
100    @staticmethod
101    def rss_inner(n):
102        g.bp[n]['rss_details'] = ds.bp_rss_details()
103        d = g.bp[n]

```

```

104
105
106 # RSS to specific values (EoS and 9 strains)
107 if(d['a0'] is not None and d['c_a0'] is not None):
108     g.bp[n]['rss_details']['a0'] = g.rss_w['a0'] * (d['a0'] - d['c_a0'])**2
109 if(d['e0'] is not None and d['c_e0'] is not None):
110     g.bp[n]['rss_details']['e0'] = g.rss_w['e0'] * (d['e0'] - d['c_e0'])**2
111 if(d['b0'] is not None and d['c_b0'] is not None):
112     g.bp[n]['rss_details']['b0'] = g.rss_w['b0'] * (d['b0'] - d['c_b0'])**2
113 if(d['ec'] is not None and d['c_ec'] is not None):
114     g.bp[n]['rss_details']['ec'] = g.rss_w['ec'] * sum(sum((d['ec'] - d['c_ec']))**2))
115
116 # Match to Rose eos over a wider parameter range
117 if(d['rose_eos'] and d['rose_e'] is not None and d['c_rose_e'] is not None):
118     g.bp[n]['rss_details']['rose'] = g.rss_w['rose'] * sum((d['rose_e'] - d['c_rose_e']))**2)
119
120 # Match to MSP, MKP cubic elastic constants over strain
121 if(d['msp_ec'] and d['msp_e'] is not None and d['c_msp_e'] is not None):
122     msp_e_zeroed = d['msp_e'][ :, :] - d['msp_e'][0, :]
123     c_msp_e_zeroed = d['c_msp_e'][ :, :] - d['c_msp_e'][0, :]
124     g.bp[n]['rss_details']['msp_shape'] = g.rss_w['msp'][0] * sum(sum((msp_e_zeroed - c_msp_e_zeroed)**2))
125     g.bp[n]['rss_details']['msp_values'] = g.rss_w['msp'][1] * sum(sum((d['msp_ec'] - d['c_msp_e']))**2))
126
127 # Fitting to BM EoS points
128 if(d['bm_eos'] and d['bm_eos_e_known_z'] is not None and d['bm_eos_e_potential_z'] is not None):
129     g.bp[n]['rss_details']['bm_eos_shape'] = g.rss_w['bm_eos'][0] * sum((d['bm_eos_e_known_z'] - d['bm_eos_e_potential_z']))**2)
130 if(d['bm_eos'] and d['bm_eos_e_known'] is not None and d['bm_eos_e_potential'] is not None):
131     g.bp[n]['rss_details']['bm_eos_values'] = g.rss_w['bm_eos'][1] * sum((d['bm_eos_e_known'] - d['bm_eos_e_potential']))**2)
132
133
134
135 # Penalties for negative values and instabilities
136 penalty = 0.0
137 if(d['c_b0'] < 0.0):
138     penalty = penalty + 1.0e3
139 for i in range(6):
140     for j in range(6):
141         if(d['c_ec'][i,j] < 0.0):
142             penalty = penalty + 1.0e2
143 if(not d['c_cubic_stability']):
144     penalty = penalty + 1.0e3
145 g.bp[n]['rss_details']['bp_penalty'] = penalty
146
147
148 # Weight
149 t = 0.0
150 for k in g.bp[n]['rss_details'].keys():
151     g.bp[n]['rss_details'][k] = g.bp[n]['weight'] * g.rss_w['bp'] * g.bp[n]['rss_details'][k]
152     t = t + g.bp[n]['rss_details'][k]
153 g.bp[n]['rss_details']['bp_total'] = t
154 g.bp[n]['rss'] = t
155 #print(g.bp[n]['rss_details'])
156 return g.bp[n]['rss']
157
158
159 """
160 Build configurations
161 #####

```

```

163
164     @staticmethod
165     def build():
166         output.log("Create BP configurations", verbose=3)
167         for n in range(len(g.bp)):
168             bp.build_inner(n)
169
170     @staticmethod
171     def build_inner(n):
172         if(g.bp[n]['built']):
173             bp.reset()
174             return False
175
176         # Get details for structure
177         structure = g.bp[n]['structure']
178         label = g.bp[n]['label']
179         a0 = g.bp[n]['a0']
180         uv = g.bp[n]['uv']
181
182         # Create original unedited structure and build neighbour list
183         rcut = g.bp_settings['eos_rcut']
184         cell_size = g.bp_settings['eos_cell_size']
185         eos_bm_original_id = configs.add_common(structure, label, a0, uv, [cell_size], rcut, 'bp', 'e')
186         g.bp[n]['eos_bm_original_id'] = eos_bm_original_id
187         nl.build_nl(g.bp[n]['eos_bm_original_id'])
188
189
190     """
191     Configs to relax and find a0 for the potential
192     #####
193
194     # Create Relax
195     g.bp[n]['relax_id'] = configs.duplicate(eos_bm_original_id)
196
197
198     """
199     BM EoS configs to calculate values E0 V0 B0 for
200     this potential
201     #####
202
203     eos_strain = g.bp_settings['eos_strain']
204     eos_steps = g.bp_settings['eos_steps']
205     eosss = 2 * eos_steps + 1
206
207     g.bp[n]['eos_ids'] = []
208     g.bp[n]['eos_strains'] = ds.bp_eos_arr(eosss)
209     g.bp[n]['eos_volumes'] = ds.bp_eos_arr(eosss)
210     g.bp[n]['eos_energies'] = ds.bp_eos_arr(eosss)
211
212     k = 0
213     for i in range(-eos_steps, eos_steps+1):
214         s = eos_strain * (i / eos_steps)
215         g.bp[n]['eos_strains'][k] = s
216         g.bp[n]['eos_ids'].append(configs.duplicate(eos_bm_original_id))
217         k = k + 1
218
219
220     """
221     ! FOR COMPUTE VALUE CALCULATIONS
222     ! Ravindran, Fast, Korzhavyi, Johansson 1998
223     ! Density functional theory for calculation of elastic properties of

```

```

224 ! orthorhombic crystals: application to TiSi2
225 !
226 #####"
227
228 rcut = g.bp_settings['ec_rcut']
229 cell_size = g.bp_settings['ec_cell_size']
230 ec_strain = g.bp_settings['ec_strain']
231 ec_steps = g.bp_settings['ec_steps']
232
233 # Create original unedited structure and build neighbour list
234 ec_rfkj_original_id = configs.add_common(structure, label, a0, uv, [cell_size], rcut, 'bp', 'e')
235 g.bp[n]['ec_rfkj_original_id'] = ec_rfkj_original_id
236 nl.build_nl(g.bp[n]['ec_rfkj_original_id'])
237
238 ecs = 2 * ec_steps + 1
239
240 g.bp[n]['ec_ids'] = []
241 g.bp[n]['ec_strains'] = ds.bp_ec_arr(ecs)
242 g.bp[n]['ec_volumes'] = ds.bp_ec_arr(ecs)
243 g.bp[n]['ec_energies'] = ds.bp_ec_arr(ecs)
244
245 g.bp[n]['c_ec'] = ds.bp_ec()
246 g.bp[n]['c_ec_gpa'] = ds.bp_ec()
247
248 for di in range(9):
249     k = 0
250     g.bp[n]['ec_ids'].append([])
251     for i in range(-ec_steps, ec_steps+1):
252         s = ec_strain * (i / ec_steps)
253         g.bp[n]['ec_strains'][di, k] = s
254         g.bp[n]['ec_ids'][di].append(configs.duplicate(ec_rfkj_original_id))
255         k = k + 1
256
257 """
258 Rose equation of state
259 Generate predicted energy vs lattice parameter data
260 points for fitting
261 (See G. Bonny exact parameter fitting code)
262 #####
263
264 if(g.bp[n]['rose_eos']):
265     # Create rose configs
266     rcut = g.bp_settings['rose_rcut']
267     steps = g.bp_settings['rose_steps']
268     cell_size = g.bp_settings['rose_cell_size']
269     a0_start = a0 + g.bp_settings['rose_a0_d1']
270     a0_mid = a0 - g.bp_settings['rose_a0_d1']
271     a0_end = a0 + g.bp_settings['rose_a0_d2']
272     xa = int(2/3 * steps)
273     xb = steps - xa
274     g.bp[n]['rose_a'] = ds.bp_rose_a_arr(steps)
275     g.bp[n]['rose_e'] = ds.bp_rose_e_arr(steps)
276     g.bp[n]['c_rose_e'] = ds.bp_rose_e_arr(steps)
277     g.bp[n]['rose_ids'] = []
278     g.bp[n]['rose_a'][::xa] = numpy.linspace(a0_start, a0_mid, xa)
279     g.bp[n]['rose_a'][xa:] = numpy.linspace(a0_mid, a0_end, xb + 1)[1:]
280
281     for i in range(steps):
282         r_a0 = g.bp[n]['rose_a'][i]
283         rose_id = configs.add_common(structure, label, r_a0, uv, [cell_size], rcut, 'bp', 'e')

```

```

285     g.bp[n]['rose_ids'].append(rose_id)
286     nl.build_nl(rose_id)
287
288     # Fill in rose computed e
289     known_a0 = g.bp[n]['a0']
290     known_e0 = g.bp[n]['e0']
291     known_v0 = g.bp[n]['v0']
292     known_b0 = g.bp[n]['b0']
293
294     # Calculate energy based on b0, a0, ecoh, vol
295     g.bp[n]['rose_e'] = rose.calc(g.bp[n]['rose_a'], known_a0, known_b0, known_e0, known_v0)
296
297
298 """
299 ! For Fitting Points
300 ! Mehl Singh Papaconstantopoulos strains 1993
301 ! Properties of ordered intermetallic alloys: first-principles
302 ! and approximate methods
303 !
304 ! Mehl Klein Papaconstantopoulos 1993
305 ! First principles calculations of elastic properties of metals
306 #####
307
308 if(g.bp[n]['msp_ec']):
309     g.bp[n]['msp_ids'] = [[],[]]
310
311     rcut = g.bp_settings['msp_rcut']
312     cell_size = g.bp_settings['msp_cell_size']
313
314     # set strains
315     g.bp[n]['msp_strains'] = numpy.linspace(0.0, g.bp_settings['msp_ec_strain'], g.bp_settings['msp_ec_steps'])
316
317     # set energy arrays
318     g.bp[n]['msp_e'] = ds.dp_msp_e_arr(g.bp_settings['msp_ec_steps'])
319     g.bp[n]['c_msp_e'] = ds.dp_msp_e_arr(g.bp_settings['msp_ec_steps'])
320
321     # C11-C12 strains
322     for i in range(g.bp_settings['msp_ec_steps']):
323         uv_ortho = ds.uv_msp_orthorhombic(g.bp[n]['msp_strains'][i])
324         msp_id = configs.add_common(structure, label, a0, uv_ortho, [cell_size], rcut, 'bp', 'e')
325         g.bp[n]['msp_ids'][0].append(msp_id)
326
327     # C44 strains
328     for i in range(g.bp_settings['msp_ec_steps']):
329         uv_mono = ds.uv_msp_monoclinic(g.bp[n]['msp_strains'][i])
330         msp_id = configs.add_common(structure, label, a0, uv_mono, [cell_size], rcut, 'bp', 'e')
331         g.bp[n]['msp_ids'][1].append(msp_id)
332
333
334     # Calculate predicted energy-strain for C11-C12 and C44
335     known_e0 = g.bp[n]['e0']
336     known_v0 = g.bp[n]['v0']
337     known_c11 = g.bp[n]['ec'][0,0]
338     known_c12 = g.bp[n]['ec'][0,1]
339     known_c44 = g.bp[n]['ec'][3,3]
340
341     # Calculate energy based on b0, a0, ecoh, vol
342     g.bp[n]['msp_e'][:,0] = msp.calc_c11_c12(g.bp[n]['msp_strains'], known_e0, known_v0, known_c11, known_c12)
343     g.bp[n]['msp_e'][:,1] = msp.calc_c44(g.bp[n]['msp_strains'], known_e0, known_v0, known_c44)
344
345

```

```

346
347
348     """
349     ! Ravindran, Fast, Korzhavyi, Johansson 1998
350     ! Density functional theory for calculation of elastic properties of
351     ! orthorhombic crystals: application to TiSi2
352     !
353     #####
354
355
356
357
358
359
360     # Mark as built
361     g.bp[n]['built'] = True
362
363     #configs.index()
364     return True
365
366     """
367
368     Reset BP Configurations
369     #####
370     @staticmethod
371     def reset():
372         output.log("Reset BP configurations", verbose=3)
373         for n in range(len(g.bp)):
374             configs.duplicate(g.bp[n]['eos_bm_original_id'], g.bp[n]['relax_id'])
375             for k in range(len(g.bp[n]['eos_ids'])):
376                 configs.duplicate(g.bp[n]['eos_bm_original_id'], g.bp[n]['eos_ids'][k])
377                 for di in range(9):
378                     for k in range(len(g.bp[n]['ec_ids'][di])):
379                         configs.duplicate(g.bp[n]['ec_rfkj_original_id'], g.bp[n]['ec_ids'][di][k])
380
381     """
382
383     Compute EoS and Elastic Constants
384     #####
385     @staticmethod
386     def calc():
387
388         output.log("Calculate BP", verbose=3)
389         for n in range(len(g.bp)):
390             bp.calc_inner(n)
391
392
393     @staticmethod
394     def calc_inner(n):
395         output.log("BP " + str(n), verbose=3)
396
397
398     # Relax
399     #####
400
401     relax_id = g.bp[n]['relax_id']
402
403
404     # Relax
405     relax.run_relax(relax_id, pmin='e', pvar=g.bp_settings['relax'])
406

```

```

407     a0 = g.configs[relax_id]['a0']
408     uv = g.configs[relax_id]['uv']
409
410     g.bp[n]['c_a0'] = a0
411     g.bp[n]['c_uv'] = uv
412
413
414     v0 = math.cellvolume(a0, uv) / (1.0 * g.configs[relax_id]['count'])
415     g.bp[n]['volrelaxed'] = v0
416
417
418     # Compute EoS
419     #####
420
421     for eos_n in range(len(g.bp[n]['eos_ids'])):
422         # Get id
423         config_id = g.bp[n]['eos_ids'][eos_n]
424
425         # Make transformation
426         s = g.bp[n]['eos_strains'][eos_n]
427         uvt = transforms.orthorhombic(uv, s)
428
429         # Change cell and calculate energy
430         nl.change_cell(config_id, a0=a0, uv=uvt, scale_uv=False)
431         configs.calc_efc(config_id)
432
433         # Save data
434         g.bp[n]['eos_volumes'][eos_n] = math.cellvolume(a0, uvt) / (1.0 * g.configs[config_id]['count'])
435         g.bp[n]['eos_energies'][eos_n] = g.configs[config_id]['c_energy'] / (1.0 * g.configs[config_id]['count'])
436
437         eos_p = eos.fit(g.bp[n]['eos_volumes'], g.bp[n]['eos_energies'])
438         g.bp[n]['c_v0'] = eos_p['v0']
439         g.bp[n]['c_e0'] = eos_p['e0']
440         g.bp[n]['c_b0'] = eos_p['b0']
441         g.bp[n]['c_b0_gpa'] = eos_p['b0_gpa']
442         g.bp[n]['c_b0p'] = eos_p['b0p']
443
444         # Make BM EoS points used to fit to the correct curve
445         if(g.bp[n]['bm_eos']):
446             # Volume arrays
447             g.bp[n]['bm_eos_v'] = numpy.linspace(g.bp[n]['eos_volumes'][0], g.bp[n]['eos_volumes'][-1], g.bp_settings['bm_eos_points'])
448
449             # Known arrays
450             g.bp[n]['bm_eos_e_known'] = numpy.zeros((g.bp_settings['bm_eos_points'], 3,), dtype=numpy.float64)
451             p = [g.bp[n]['v0'], g.bp[n]['e0'], g.bp[n]['b0'], 2.0]
452             g.bp[n]['bm_eos_e_known'] = eos.bm_calc(p, g.bp[n]['bm_eos_v'])
453
454             # Based on potential
455             g.bp[n]['bm_eos_e_potential'] = numpy.zeros((g.bp_settings['bm_eos_points'], 3,), dtype=numpy.float64)
456             p = [g.bp[n]['c_v0'], g.bp[n]['c_e0'], g.bp[n]['c_b0'], 2.0]
457             g.bp[n]['bm_eos_e_potential'] = eos.bm_calc(p, g.bp[n]['bm_eos_v'])
458
459
460             # Known arrays
461             g.bp[n]['bm_eos_v_known_z'] = numpy.linspace(0.98 * g.bp[n]['v0'], 1.02 * g.bp[n]['v0'], g.bp_settings['bm_eos_points'])
462             p = [g.bp[n]['v0'], g.bp[n]['e0'], g.bp[n]['b0'], 2.0]
463             g.bp[n]['bm_eos_e_known_z'] = eos.bm_calc(p, g.bp[n]['bm_eos_v_known_z'])
464             g.bp[n]['bm_eos_e_known_z'] = g.bp[n]['bm_eos_e_known_z'] - g.bp[n]['e0']

```

```

466 g.bp[n]['bm_eos_v_known_z'] = g.bp[n]['bm_eos_v_known_z'] - g.bp[n]['v0']
467
468
469 # Known arrays
470 g.bp[n]['bm_eos_v_potential_z'] = numpy.linspace(0.98 * g.bp[n]['c_v0'], 1.02 * g.bp[n]['c_v0'], g.
471     bp_settings['bm_eos_points'])
472 p = [g.bp[n]['c_v0'], g.bp[n]['c_e0'], g.bp[n]['c_b0'], 2.0]
473 g.bp[n]['bm_eos_e_potential_z'] = eos.bm_calc(p, g.bp[n]['bm_eos_v_potential_z'])
474 g.bp[n]['bm_eos_e_potential_z'] = g.bp[n]['bm_eos_e_potential_z'] - g.bp[n]['c_e0']
475 g.bp[n]['bm_eos_v_potential_z'] = g.bp[n]['bm_eos_v_potential_z'] - g.bp[n]['c_v0']
476
477
478
479
480
481 # Compute EC
482 ######
483
484
485 if(g.bp[n]['rfkj_ec']):
486     for di in range(len(g.bp[n]['ec_ids'])):
487         for i in range(len(g.bp[n]['ec_ids'][0])):
488             # Get id
489             config_id = g.bp[n]['ec_ids'][di][i]
490
491             # Make transformation
492             s = g.bp[n]['ec_strains'][di, i]
493             uvt = transforms.dn(uv, s, di + 1)
494
495             # Change cell and calculate energy
496             nl.change_cell(config_id, a0=a0, uv=uvt, scale_uv=False)
497             configs.calc_efs(config_id)
498
499             # Save data
500             g.bp[n]['ec_volumes'][di, i] = a0**3 * math.tripleproduct(uvt[:,0], uvt[:,1], uvt[:,2]) / g.configs[
501                 config_id]['count']
502             g.bp[n]['ec_energies'][di, i] = g.configs[config_id]['c_energy'] / g.configs[config_id]['count']
503
504             # Distortion data
505             d = ds.bp_d()
506             for di in range(len(g.bp[n]['ec_ids'])):
507                 p = numpy.polyfit(g.bp[n]['ec_strains'][di, :], g.bp[n]['ec_energies'][di, :], 2)
508                 d[di] = p[0]
509
510             A = numpy.zeros((9,9,), dtype=numpy.float64)
511             A[0,0] = v0/2
512             A[1,1] = v0/2
513             A[2,2] = v0/2
514             A[3,3] = 2*v0
515             A[4,4] = 2*v0
516             A[5,5] = 2*v0
517             A[6,6] = -1.0*v0
518             A[7,7] = -1.0*v0
519             A[8,8] = -1.0*v0
520             A[6,0] = v0 / 2
521             A[6,1] = v0 / 2
522             A[7,0] = v0 / 2
523             A[7,2] = v0 / 2
524             A[8,1] = v0 / 2
525             A[8,2] = v0 / 2

```

```

525 C = numpy.asarray(d, dtype=numpy.float64)
526 A_inv = numpy.linalg.inv(A)
527 B = numpy.matmul(A_inv, C)
528
529 g.bp[n]['c_ec'][0,0] = B[0]
530 g.bp[n]['c_ec'][1,1] = B[1]
531 g.bp[n]['c_ec'][2,2] = B[2]
532 g.bp[n]['c_ec'][3,3] = B[3]
533 g.bp[n]['c_ec'][4,4] = B[4]
534 g.bp[n]['c_ec'][5,5] = B[5]
535 g.bp[n]['c_ec'][0,1] = B[6]
536 g.bp[n]['c_ec'][1,0] = B[6]
537 g.bp[n]['c_ec'][0,2] = B[7]
538 g.bp[n]['c_ec'][2,0] = B[7]
539 g.bp[n]['c_ec'][1,2] = B[8]
540 g.bp[n]['c_ec'][2,1] = B[8]
541
542 # Calculate in GPA
543 g.bp[n]['c_ec_gpa'] = units.convert('EV/ANG3', 'GPA', g.bp[n]['c_ec'])
544
545
546 # Compliance tensor
547 g.bp[n]['c_sc'] = math.minverse(g.bp[n]['c_ec'])
548
549 # Properties from Elastic Constants
550 g.bp[n]['c_b0_r'] = bp.bulk_r(g.bp[n]['c_sc'])
551 g.bp[n]['c_b0_v'] = bp.bulk_v(g.bp[n]['c_ec'])
552 g.bp[n]['c_b0_avg'] = 0.5 * (g.bp[n]['c_b0_r'] + g.bp[n]['c_b0_v'])
553 g.bp[n]['c_g_r'] = bp.shear_g_r(g.bp[n]['c_sc'])
554 g.bp[n]['c_g_v'] = bp.shear_g_v(g.bp[n]['c_ec'])
555 g.bp[n]['c_g_avg'] = 0.5 * (g.bp[n]['c_g_r'] + g.bp[n]['c_g_v'])
556 g.bp[n]['c_e'] = bp.youngs_e(g.bp[n]['c_b0_avg'], g.bp[n]['c_g_avg'])
557 g.bp[n]['c_melting_temperature'] = bp.melting_temperature(g.bp[n]['c_ec'])
558 g.bp[n]['c_cubic_stability'] = bp.cubic_stability(g.bp[n]['c_ec'])
559 g.bp[n]['c_stability'] = bp.stability(g.bp[n]['c_ec'])
560
561 # Convert to GPA
562 g.bp[n]['c_b0_r_gpa'] = units.convert('EV/ANG3', 'GPA', g.bp[n]['c_b0_r'])
563 g.bp[n]['c_b0_v_gpa'] = units.convert('EV/ANG3', 'GPA', g.bp[n]['c_b0_v'])
564 g.bp[n]['c_b0_avg_gpa'] = units.convert('EV/ANG3', 'GPA', g.bp[n]['c_b0_avg'])
565 g.bp[n]['c_g_r_gpa'] = units.convert('EV/ANG3', 'GPA', g.bp[n]['c_g_r'])
566 g.bp[n]['c_g_v_gpa'] = units.convert('EV/ANG3', 'GPA', g.bp[n]['c_g_v'])
567 g.bp[n]['c_g_avg_gpa'] = units.convert('EV/ANG3', 'GPA', g.bp[n]['c_g_avg'])
568 g.bp[n]['c_e_gpa'] = units.convert('EV/ANG3', 'GPA', g.bp[n]['c_e'])
569
570
571 """
572 Rose equation of state
573 Run calculations to compare to known
574 energy-lattice parameter points
575 #####
576
577
578
579 if(g.bp[n]['rose_eos']):
580     for i in range(len(g.bp[n]['rose_ids'])):
581         rose_id = g.bp[n]['rose_ids'][i]
582         configs.calc_efc(rose_id)
583         g.bp[n]['c_rose_e'][i] = g.configs[rose_id]['c_energy'] / g.configs[rose_id]['count']
584
585

```

```

586
587
588
589
590     """
591
592     MSP MKP strains C11-C12, C44
593     Run calculations to compare to known
594     energy-lattice parameter points
595     #####"
596
597     if(g.bp[n]['msp_ec']):
598         # C11-C12
599         for i in range(len(g.bp[n]['msp_ids'][0])):
600             msp_id = g.bp[n]['msp_ids'][0][i]
601             configs.calc_efs(msp_id)
602             g.bp[n]['c_msp_e'][i,0] = g.configs[msp_id]['c_energy'] / g.configs[msp_id]['count']
603         # C44
604         for i in range(len(g.bp[n]['msp_ids'][1])):
605             msp_id = g.bp[n]['msp_ids'][1][i]
606             configs.calc_efs(msp_id)
607             g.bp[n]['c_msp_e'][i,1] = g.configs[msp_id]['c_energy'] / g.configs[msp_id]['count']
608
609     # Array to store temporary results
610     ec_temp = numpy.zeros((3,), dtype=numpy.float64)
611
612     # Fit polynomial for c11_c12
613     s_c11_c12 = g.bp[n]['msp_strains'][:,]
614     e_c11_c12 = g.bp[n]['c_msp_e'][:,0]
615     p_c11_c12 = numpy.polyfit(s_c11_c12, e_c11_c12, 2)
616
617     # Fit polynomial for c44
618     s_c44 = g.bp[n]['msp_strains'][:,]
619     e_c44 = g.bp[n]['c_msp_e'][:,1]
620     p_c44 = numpy.polyfit(s_c44, e_c44, 2)
621
622     # Get bulk modulus and
623     b0 = g.bp[n]['c_b0']
624     v0 = g.bp[n]['c_v0']
625     if(v0 == 0.0):
626         v0 = 1.0e-20
627
628     # Solve
629     a_mat= numpy.zeros((2,2,), dtype=numpy.float64)
630     a_mat[0,0] = 1.0
631     a_mat[0,1] = -1.0
632     a_mat[1,0] = 1.0/3.0
633     a_mat[1,1] = 2.0/3.0
634     b_mat= numpy.zeros((2,), dtype=numpy.float64)
635     b_mat[0] = p_c11_c12[0] / v0
636     b_mat[1] = b0
637     c_mat = numpy.linalg.solve(a_mat, b_mat)
638
639     ec_temp[0] = c_mat[0]
640     ec_temp[1] = c_mat[1]
641     ec_temp[2] = (2 * p_c44[0]) / v0
642
643     g.bp[n]['c_msp_ec'] = ds.bp_ec(ec_temp)
644     g.bp[n]['c_msp_ec_gpa'] = units.convert('EV/ANG3', 'GPA', g.bp[n]['c_msp_ec'])
645
646

```

```

647     output.log("", verbose=3)
648     output.log("BP Results " + str(n), verbose=3)
649     output.log("====", verbose=3)
650     output.log("a0    " + str(g.bp[n]['c_a0']), verbose=3)
651     output.log("uv    " + str(g.bp[n]['c_uv'][0,0]) + " " + str(g.bp[n]['c_uv'][0,1]) + " " + str(g.bp[n]['c_uv'
       ][0,2]), verbose=3)
652     output.log("      " + str(g.bp[n]['c_uv'][1,0]) + " " + str(g.bp[n]['c_uv'][1,1]) + " " + str(g.bp[n]['c_uv'
       ][1,2]), verbose=3)
653     output.log("      " + str(g.bp[n]['c_uv'][2,0]) + " " + str(g.bp[n]['c_uv'][2,1]) + " " + str(g.bp[n]['c_uv'
       ][2,2]), verbose=3)
654     output.log("v0    " + str(g.bp[n]['c_v0']), verbose=3)
655     output.log("e0    " + str(g.bp[n]['c_e0']), verbose=3)
656     output.log("b0    " + str(g.bp[n]['c_b0']), verbose=3)
657     output.log("b0gpa " + str(g.bp[n]['c_b0_gpa']), verbose=3)
658     output.log("", verbose=3)
659     output.log("", verbose=3)
660
661
662 @staticmethod
663 def bulk_r(sc):
664     return 1.0 / (sc[0,0] + sc[1,1] + sc[2,2] + 2.0 * (sc[0,1] + sc[0,2] + sc[1,2]))
665
666 @staticmethod
667 def bulk_v(ec):
668     return ((1.0 / 9.0) * (ec[0,0] + ec[1,1] + ec[2,2]) + (2.0 / 9.0) * (ec[0,1] + ec[0,2] + ec[1,2]))
669
670
671 @staticmethod
672 def shear_g_r(sc):
673     return (15.0 / (4*(sc[0,0]+sc[1,1]+sc[2,2]) - 4*(sc[0,1]+sc[0,2]+sc[1,2]) + 3* (sc[3,3]+sc[4,4]+sc[5,5])))
674
675 @staticmethod
676 def shear_g_v(ec):
677     return (1.0/15.0) * (ec[0,0]+ec[1,1]+ec[2,2]-ec[0,1]-ec[0,2]-ec[1,2]) + (1.0/5.0) * (ec[3,3] + ec[4,4] + ec
       [5,5])
678
679 @staticmethod
680 def shear_g_vec(sc):
681     gvec = numpy.zeros((3,), dtype=numpy.float64)
682     gvec[0] = 1.0 / (2.0 * sc[3,3])
683     gvec[1] = 1.0 / (2.0 * sc[4,4])
684     gvec[2] = 1.0 / (2.0 * sc[5,5])
685     return gvec
686
687 @staticmethod
688 def youngs_e(b0_avg, g_avg):
689     return (9 * b0_avg * g_avg) / (3 * b0_avg + g_avg)
690
691 @staticmethod
692 def shear_e_vec(sc):
693     evec = numpy.zeros((3,), dtype=numpy.float64)
694     evec[0] = 1.0 / sc[0,0]
695     evec[1] = 1.0 / sc[1,1]
696     evec[2] = 1.0 / sc[2,2]
697     return evec
698
699 @staticmethod
700 def melting_temperature(ec):
701     return 598.0 + 6.66 * (ec[0,0]+ec[1,1]+ec[2,2]) - 0.003 * (ec[0,0]+ec[1,1]+ec[2,2])**2
702
703

```

```

704     @staticmethod
705     def cubic_stability(ec):
706         if((ec[0,0] + 2 * ec[0,1]) > 0 and (ec[0,0] - ec[0,1]) > 0 and ec[3,3] > 0):
707             return True
708         return False
709
710
711     @staticmethod
712     def stability(ec):
713         if((ec[0,0] > 0.0) and (ec[1,1] > 0.0) and (ec[2,2] > 0.0) and
714             (ec[3,3] > 0.0) and (ec[4,4] > 0.0) and (ec[5,5] > 0.0) and
715             (ec[0,0] + ec[1,1] - 2 * ec[0,1]) > 0.0 and
716             (ec[0,0] + ec[2,2] - 2 * ec[0,2]) > 0.0 and
717             (ec[1,1] + ec[2,2] - 2 * ec[1,2]) > 0.0 and
718             (ec[0,0] + ec[1,1] + ec[2,2] + 2 * ec[0,1] + 2 * ec[0,2] + 2 * ec[1,2]) > 0):
719                 return True
720             return False

```

S.7 Interpolation

Lagrange polynomial interpolation was programmed in Fortran and was used in a number of places throughout the program (listing S.10).

Listing S.10: Lagrange polynomial interpolation

```

1 SUBROUTINE interpn(xi, x, y, yi)
2 ! INTERP N SIZES DATA ARRAY
3 IMPLICIT NONE
4 ! IN/OUT
5 REAL(kind=DoubleReal), INTENT(IN) :: xi
6 REAL(kind=DoubleReal), INTENT(IN) :: x(:)
7 REAL(kind=DoubleReal), INTENT(IN) :: y(:)
8 REAL(kind=DoubleReal), INTENT(OUT) :: yi
9 ! PRIVATE
10 INTEGER(kind=StandardInteger) :: i, j, n
11 REAL(kind=DoubleReal) :: li
12 !#####
13 n = SIZE(x,1)
14 yi = 0.0D0
15 IF (SIZE(x,1) .EQ. SIZE(y,1)) THEN
16     DO i = 1, n
17         li = 1.0D0
18         DO j = 1, n
19             IF(i .NE. j) THEN
20                 li = li * (xi - x(j)) / (x(i) - x(j))
21             END IF
22         END DO
23         yi = yi + li * y(i)
24     END DO
25 END IF
26 !#####
27 END SUBROUTINE interpn

```

S.8 Interpolation (Gradient)

A modified version of the Lagrange polynomial interpolation was programmed in Fortran to interpolate the derivative of a set of data points at a given position between (and including) the start and end point (listing

S.11).

Listing S.11: Lagrange polynomial interpolation (Gradients)

```

1 SUBROUTINE interpndydx(xi, x, y, ypi)
2 ! Interpolate and return derivative at xi
3 IMPLICIT NONE
4 ! IN/OUT
5 REAL(kind=DoubleReal), INTENT(IN) :: xi
6 REAL(kind=DoubleReal), INTENT(IN) :: x(:)
7 REAL(kind=DoubleReal), INTENT(IN) :: y(:)
8 REAL(kind=DoubleReal), INTENT(OUT) :: ypi
9 ! PRIVATE
10 INTEGER(kind=StandardInteger) :: i, j, k, n
11 REAL(kind=DoubleReal) :: fx, gx, psum
12 !#####
13 n = SIZE(x,1)
14 IF (SIZE(x,1) .EQ. SIZE(y,1)) THEN
15   ypi = 0.0D0
16   Do i=1,SIZE(x,1)
17     fx = 1.0D0
18     gx = 0.0D0
19     Do j=1,SIZE(x,1)
20       If(i .NE. j) Then
21         fx = fx / (x(i) - x(j))
22         psum = 1.0D0
23         Do k=1,SIZE(x,1)
24           If((i .NE. k) .AND. (j .NE. k))Then
25             psum = psum * (xi - x(k))
26           End If
27         End Do
28         gx = gx + psum
29       End If
30     End Do
31     ypi = ypi + fx * gx * y(i)
32   End Do
33 END IF
34 !#####
35 END SUBROUTINE interpndydx

```

Appendix T

Potential Functions

T.1 Introduction

This section of the appendix covers the types of potential function and common choices of function used and details on how to use these in the potential fitting code.

T.2 Functions

T.2.1 Ackland Embedding

Listing T.1: Ackland Embedding Fortran code

```
1  !# Olsson/Wallenius
2  !# Ackland Mendelev 2004
3
4  SUBROUTINE ackland_embedding(r, p, p_fixed, y)
5  !#####
6  ! f(x) = -sqrt(r) + A * r**2 + B * r**4
7  IMPLICIT NONE
8  !#####
9  REAL(kind=DoubleReal), INTENT(IN) :: r
10 REAL(kind=DoubleReal), INTENT(IN) :: p(:)
11 REAL(kind=DoubleReal), INTENT(IN) :: p_fixed(:)
12 REAL(kind=DoubleReal), INTENT(OUT) :: y
13 !#####
14 !#####
15 IF(r .LT. 0.0D0)THEN
16     y = 0.0D0
17 ELSE
18     y = -1.0D0 * sqrt(r) + p(1) * (r)**2 + p(2) * (r)**4
19 END IF
20 END SUBROUTINE ackland_embedding
21
22
23
24 SUBROUTINE ackland_embedding_v(r, p, p_fixed, y)
25 !#####
26 ! f(x) = A * sqrt(r) + B * r**2 + C * r**4
27 IMPLICIT NONE
28 !#####
29 REAL(kind=DoubleReal), INTENT(IN) :: r(:)
```

```

30 REAL(kind=DoubleReal), INTENT(IN) :: p(:)
31 REAL(kind=DoubleReal), INTENT(IN) :: p_fixed(:)
32 REAL(kind=DoubleReal), INTENT(OUT) :: y(1:SIZE(r,1))
33 INTEGER(kind=StandardInteger) :: n
34 !#####
35 DO n = 1, SIZE(r,1)
36   CALL ackland_embedding(r(n), p, p_fixed, y(n))
37 END DO
38 END SUBROUTINE ackland_embedding_v

```

Listing T.2: Ackland Embedding EAMPA input file

```

1 #TYPE ackland_embedding
2 #P -0.5 1.0
3 #PF 0.0

```

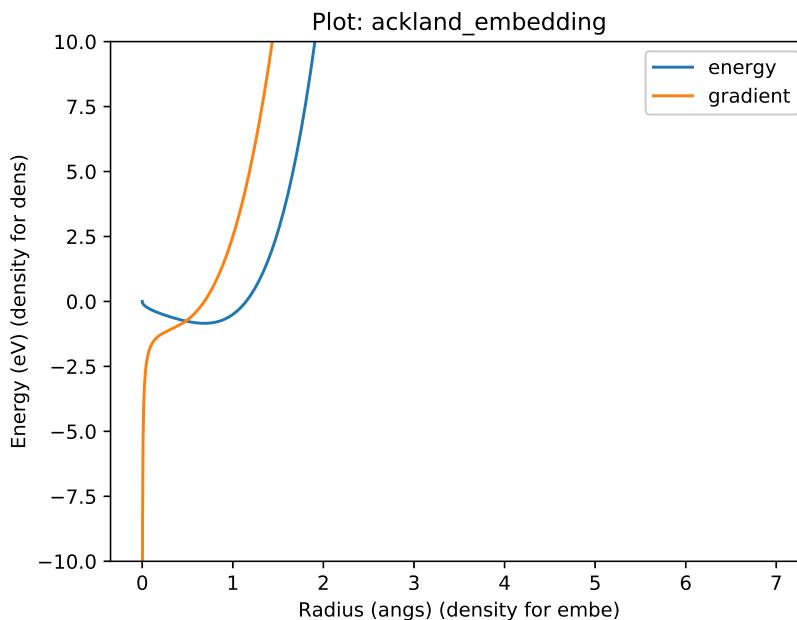


Figure T.1: Ackland Mendelev

T.2.2 Buckingham

Listing T.3: Buckingham Fortran code

```

1 ! Buckingham Potential
2
3 SUBROUTINE buckingham(r, p, p_fixed, y)
4 !#####
5 ! BUCKINGHAM POTENTIAL
6 IMPLICIT NONE
7 !#####
8 REAL(kind=DoubleReal), INTENT(IN) :: r
9 REAL(kind=DoubleReal), INTENT(IN) :: p(1:3)
10 REAL(kind=DoubleReal), INTENT(IN) :: p_fixed(1:1)
11 REAL(kind=DoubleReal), INTENT(OUT) :: y
12 !#####
13 y = p(1) * exp(-1 * p(2) * r) - p(3) / r**6
14 END SUBROUTINE buckingham
15

```

```

16
17
18 SUBROUTINE buckingham_v(r, p, p_fixed, y)
19 !#####
20 ! BUCKINGHAM POTENTIAL
21 IMPLICIT NONE
22 !#####
23 REAL(kind=DoubleReal), INTENT(IN) :: r(:)
24 REAL(kind=DoubleReal), INTENT(IN) :: p(1:3)
25 REAL(kind=DoubleReal), INTENT(IN) :: p_fixed(1:1)
26 REAL(kind=DoubleReal), INTENT(OUT) :: y(1:SIZE(r,1))
27 !#####
28 y(:) = p(1) * exp(-1 * p(2) * r(:)) - p(3) / r(:)**6
29 END SUBROUTINE buckingham_v

```

Listing T.4: Buckingham EAMPA input file

```

1 #TYPE buckingham
2 #P 6.0 0.5 12.0
3 #PF 0.0

```

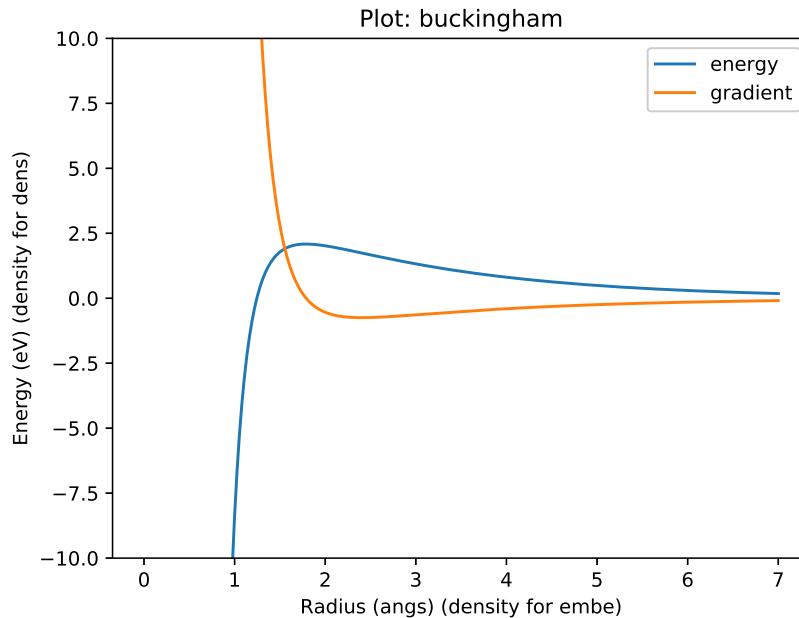


Figure T.2: Buckingham

T.2.3 Lennard Jones

Listing T.5: Lennard Jones Fortran code

```

1
2
3 SUBROUTINE lennard_jones(r, p, p_fixed, y)
4 !#####
5 ! LENNARD JONES FUNCTION
6 IMPLICIT NONE
7 !#####
8 REAL(kind=DoubleReal), INTENT(IN) :: r
9 REAL(kind=DoubleReal), INTENT(IN) :: p(1:2)
10 REAL(kind=DoubleReal), INTENT(IN) :: p_fixed(1:1)

```

```

11 REAL(kind=DoubleReal), INTENT(OUT) :: y
12 !#####
13 y = p(1) * ((p(2) / r)**12 - 2 * (p(2)/r)**6)
14 END SUBROUTINE lennard_jones
15
16 SUBROUTINE lennard_jones_v(r, p, p_fixed, y)
17 !#####
18 ! LENNARD JONES VECTOR FUNCTION
19 IMPLICIT NONE
20 !#####
21 REAL(kind=DoubleReal), INTENT(IN) :: r(:)
22 REAL(kind=DoubleReal), INTENT(IN) :: p(1:2)
23 REAL(kind=DoubleReal), INTENT(IN) :: p_fixed(1:1)
24 REAL(kind=DoubleReal), INTENT(OUT) :: y(1:SIZE(r,1))
25 !#####
26 INTEGER(kind=StandardInteger) :: n
27 !#####
28 DO n = 1, SIZE(r,1)
29   CALL lennard_jones(r(n), p, p_fixed, y(n))
30 END DO
31 END SUBROUTINE lennard_jones_v

```

Listing T.6: Lennard Jones EAMPA input file

```

1 #TYPE lennard_jones
2 #P    2.3 3.5
3 #PF   0.0

```

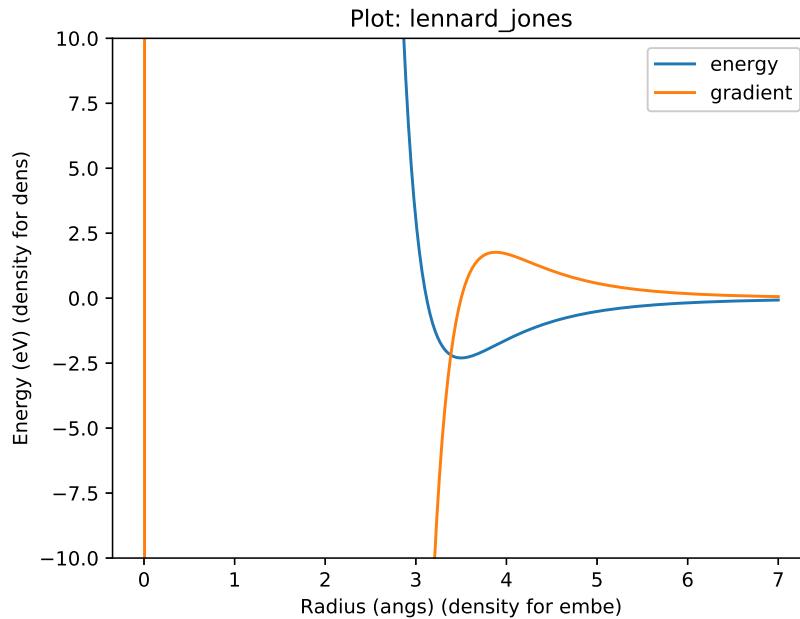


Figure T.3: Lennard Jones

T.2.4 Mishin Density

Listing T.7: Mishin Density Fortran code

```

1 !# Mishin
2
3 SUBROUTINE mishin_density(r, p, p_fixed, y)

```

```

4 !#####
5 ! f(x) = (N r^3 e^{-(z r)})^2
6 IMPLICIT NONE
7 !#####
8 REAL(kind=DoubleReal), INTENT(IN) :: r
9 REAL(kind=DoubleReal), INTENT(IN) :: p(1:7)
10 REAL(kind=DoubleReal), INTENT(IN) :: p_fixed(1:1)
11 REAL(kind=DoubleReal), INTENT(OUT) :: y
12 !#####
13 REAL(kind=DoubleReal) :: rc, r0, h
14 REAL(kind=DoubleReal) :: A, B, C, gamma, yb
15 REAL(kind=DoubleReal) :: psi, e, x, z
16 !#####
17 rc = p_fixed(1)
18 r0 = p(1)
19 h = p(2)
20 A = p(3)
21 B = p(4)
22 C = p(5)
23 gamma = p(6)
24 yb = p(7)
25
26 IF(r .GT. rc)THEN
27   y = 0.0D0
28 ELSE
29   x = ((r - rc) / h)**4
30   psi = x / (1 + x)
31   z = r - r0
32   e = exp(-1.0D0 * gamma * z)
33   y = psi * (A * z**yb * e * (1 + B * e) + C)
34 END IF
35 END SUBROUTINE mishin_density
36
37
38 SUBROUTINE mishin_density_v(r, p, p_fixed, y)
39 !#####
40 ! f(x) = (N r^3 e^{-(z r)})^2
41 IMPLICIT NONE
42 !#####
43 REAL(kind=DoubleReal), INTENT(IN) :: r(:)
44 REAL(kind=DoubleReal), INTENT(IN) :: p(:)
45 REAL(kind=DoubleReal), INTENT(IN) :: p_fixed(:)
46 REAL(kind=DoubleReal), INTENT(OUT) :: y(1:SIZE(r,1))
47 INTEGER(kind=StandardInteger) :: n
48 !#####
49 DO n = 1, SIZE(r,1)
50   CALL mishin_density(r(n), p, p_fixed, y(n))
51 END DO
52 END SUBROUTINE mishin_density_v

```

Listing T.8: Mishin Density EAMPA input file

```

1 #TYPE mishin_density
2 #P      0.5 0.3 1.0 1.0 1.0 &
3      1.0 1.0
4 #PF     6.0

```

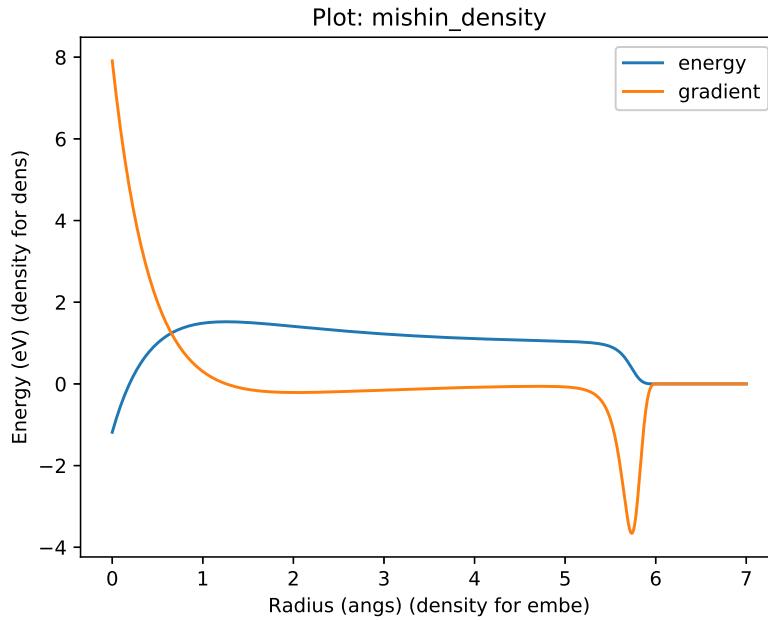


Figure T.4: Mishin Density

T.2.5 Morse

Listing T.9: Morse Fortran code

```

1
2
3 SUBROUTINE morse(r, p, p_fixed, y)
4 !#####
5 ! MORSE FUNCTION
6 IMPLICIT NONE
7 !#####
8 REAL(kind=DoubleReal), INTENT(IN) :: r
9 REAL(kind=DoubleReal), INTENT(IN) :: p(1:3)
10 REAL(kind=DoubleReal), INTENT(IN) :: p_fixed(1:1)
11 REAL(kind=DoubleReal), INTENT(OUT) :: y
12 !#####
13 y = p(1) * (exp(-2.0D0 * p(2) * (r - p(3))) - 2.0D0 * exp(-p(2)*(r - p(3))))
14 END SUBROUTINE morse
15
16
17
18
19 SUBROUTINE morse_v(r, p, p_fixed, y)
20 !#####
21 ! MORSE FUNCTION
22 IMPLICIT NONE
23 !#####
24 REAL(kind=DoubleReal), INTENT(IN) :: r(:)
25 REAL(kind=DoubleReal), INTENT(IN) :: p(1:3)
26 REAL(kind=DoubleReal), INTENT(IN) :: p_fixed(1:1)
27 REAL(kind=DoubleReal), INTENT(OUT) :: y(1:SIZE(r,1))
28 !#####
29 y(:) = p(1) * (exp(-2.0D0 * p(2) * (r(:) - p(3))) - 2.0D0 * exp(-p(2)*(r(:) - p(3))))
30 END SUBROUTINE morse_v

```

Listing T.10: Mishin Density EAMPA input file

```

1 #TYPE morse
2 #P    4.669 1.256 2.8
3 #PF   0.0

```

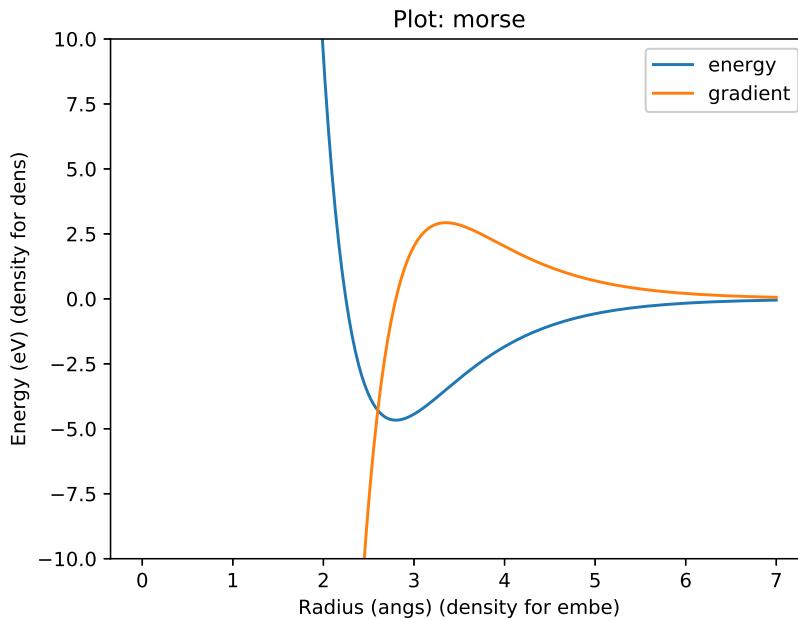


Figure T.5: Morse

T.2.6 Polynomial Embedding

Listing T.11: Polynomial Embedding Fortran code

```

1  !# Polynomial embedding
2  !# General function for embedding functions
3  !# like F(r) = sqrt(r)
4  !#     F(r) = A sqrt(r) + B r**2 + C r**3
5
6
7  SUBROUTINE polynomial_embedding(r, p, p_fixed, y)
8  !#####
9  ! f(x) = (N r^3 e^(-z r))^2
10 !IMPLICIT NONE
11 !#####
12 REAL(kind=DoubleReal), INTENT(IN) :: r
13 REAL(kind=DoubleReal), INTENT(IN) :: p(:)      !# Coeff
14 REAL(kind=DoubleReal), INTENT(IN) :: p_fixed(:) !# Power
15 REAL(kind=DoubleReal), INTENT(OUT) :: y
16 !#####
17 INTEGER(kind=StandardInteger) :: n
18 !#####
19 y = 0.0D0
20
21 IF(SIZE(p,1) .NE. SIZE(p_fixed,1))THEN
22   RETURN
23 END IF
24
25 DO n = 1, SIZE(p,1)
26   y = y + p(n) * r**p_fixed(n)

```

```

27 END DO
28
29 END SUBROUTINE polynomial_embedding
30
31
32
33 SUBROUTINE polynomial_embedding_v(r, p, p_fixed, y)
34 !#####
35 !
36 IMPLICIT NONE
37 !#####
38 REAL(kind=DoubleReal), INTENT(IN) :: r(:)
39 REAL(kind=DoubleReal), INTENT(IN) :: p(:)
40 REAL(kind=DoubleReal), INTENT(IN) :: p_fixed(:)
41 REAL(kind=DoubleReal), INTENT(OUT) :: y(1:SIZE(r,1))
42 INTEGER(kind=StandardInteger) :: n
43 !#####
44 DO n = 1, SIZE(r,1)
45   CALL polynomial_embedding(r(n), p, p_fixed, y(n))
46 END DO
47 END SUBROUTINE polynomial_embedding_v

```

Listing T.12: Polynomial Embedding EAMPA input file

```

1 #TYPE polynomial_embedding
2 #P &
3 0.038121933276638625 -1.1405198957613036 0.5042566260031044
4 #PF &
5 0.0 0.5 2.0

```

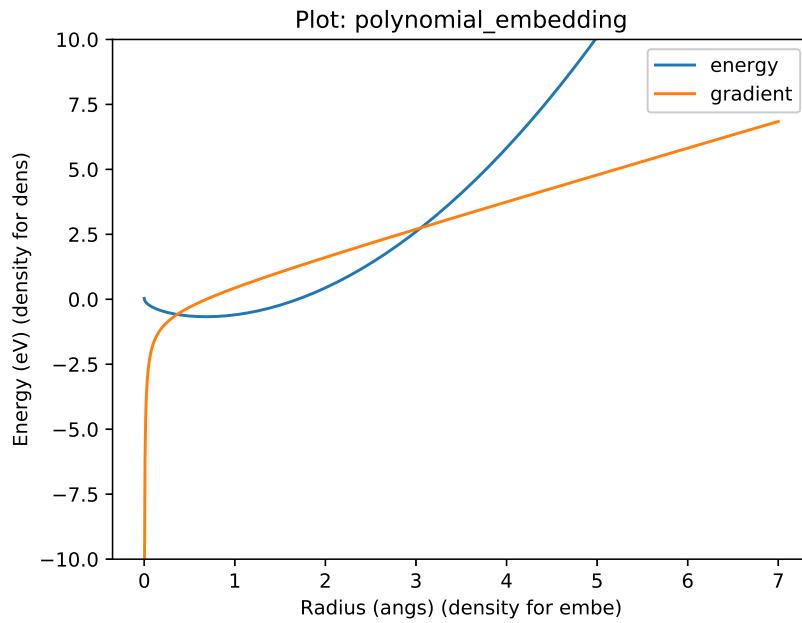


Figure T.6: Polynomial Embedding

T.2.7 Spline

Listing T.13: Spline Fortran code

```

1 ! POWER SPLINE

```

```

2 ! sum (ai (r - ri)^3 H(ri - r)) - cubic example
3 ! sum (ai (r - ri)^5 H(ri - r)) - quintic example
4
5 ! SCALAR SUBROUTINE
6 SUBROUTINE spline(r, p, p_fixed, y)
7 !#####
8 ! p coefficients
9 ! pf r cutoffs
10 ! they must be the same size
11 IMPLICIT NONE
12 !#####
13 REAL(kind=DoubleReal), INTENT(IN) :: r
14 REAL(kind=DoubleReal), INTENT(IN) :: p(:)
15 REAL(kind=DoubleReal), INTENT(IN) :: p_fixed(:)
16 REAL(kind=DoubleReal), INTENT(OUT) :: y
17 !#####
18 INTEGER(kind=StandardInteger) :: ps, pfs, ss
19 REAL(kind=DoubleReal) :: settings(1:10)
20 REAL(kind=DoubleReal) :: nodes(1:SIZE(p,1))
21 REAL(kind=DoubleReal) :: rk
22 INTEGER(kind=StandardInteger) :: n, k
23 ! Settings
24 INTEGER(kind=StandardInteger) :: power_lower
25 INTEGER(kind=StandardInteger) :: power_upper
26 INTEGER(kind=StandardInteger) :: rkoption
27 REAL(kind=DoubleReal) :: zbl_qa
28 REAL(kind=DoubleReal) :: zbl_qb
29 REAL(kind=DoubleReal) :: h_node_r = 0.0D0
30 REAL(kind=DoubleReal) :: h_node_c = 0.0D0
31 INTEGER(kind=StandardInteger) :: h_node_p
32 REAL(kind=DoubleReal) :: yzbl
33 !#####
34 y = 0.0D0
35 ss = 10
36 ps = SIZE(p,1)
37 pfs = SIZE(p_fixed,1)
38
39 IF(pfs .LT. ss)THEN
40   RETURN
41 END IF
42
43 !Read Settings
44 settings(1:ss) = p_fixed(1:ss)
45 power_lower = INT(settings(1)) ! 3, 5 etc
46 power_upper = INT(settings(2)) !
47 rkoption = INT(settings(3)) ! 0 = (rk - r)^n 1 = (r - rk)^n
48 zbl_qa = settings(4)
49 zbl_qb = settings(5)
50 h_node_r = settings(6) ! Hardening node rcutoff
51 h_node_c = settings(7) ! Hardening node coefficient
52 h_node_p = INT(settings(8)) ! Hardening node power
53
54 IF(pfs .NE. (ps + ss))THEN
55   RETURN
56 END IF
57 nodes(1:ps) = p_fixed(ss+1:pfs)
58
59 IF(r .GT. nodes(ps))THEN
60   RETURN
61 END IF

```

```

63
64 ! Hardening node (See G. Bonny 2012 code)
65 IF(h_node_r .GT. 0.0D0 .AND. r .LE. h_node_r)Then
66   y = y + h_node_c * (h_node_r - r)**h_node_p
67 END IF
68
69 ! Power spline
70 k = power_lower
71 DO n = 1, ps
72   rk = nodes(n)
73   IF((rk - r) .GE. 0.0D0)THEN
74     IF(rkoption .EQ. 0)THEN
75       y = y + p(n) * (rk - r)**k
76     ELSE
77       y = y + p(n) * (r - rk)**k
78     END IF
79   END IF
80   IF(k .LT. power_upper)THEN
81     k = k + 1
82   ELSE
83     k = power_lower
84   END IF
85 END DO
86
87 ! Add ZBL
88 IF(zbl_qa .GT. 0.0D0 .AND. zbl_qb .GT. 0.0D0)THEN
89   CALL fzbl(r, zbl_qa, zbl_qb, yzbl)
90   y = y + yzbl
91 END IF
92
93
94
95
96
97
98 END SUBROUTINE spline
99
100
101 ! VECTOR SUBROUTINE
102 SUBROUTINE spline_v(r, p, p_fixed, y)
103 !#####
104 ! CUBIC SPLINE
105 IMPLICIT NONE
106 !#####
107 REAL(kind=DoubleReal), INTENT(IN) :: r(:)
108 REAL(kind=DoubleReal), INTENT(IN) :: p(:)
109 REAL(kind=DoubleReal), INTENT(IN) :: p_fixed(:)
110 REAL(kind=DoubleReal), INTENT(OUT) :: y(1:SIZE(r,1))
111 INTEGER(kind=StandardInteger) :: n
112 !#####
113 ! Loop through all the values in r(:), calculate and store in y(:)
114 DO n = 1, SIZE(r,1)
115   CALL spline(r(n), p, p_fixed, y(n))
116 END DO
117
118 END SUBROUTINE spline_v

```

Listing T.14: Spline EAMPA input file

```

1 #TYPE spline
2 #P &
3 -0.02824709259353992 -0.0038975302342977554 0.009294738388828702 3.9855208652451597 0.0028557787639382896&

```

```

4 0.0018722262741587487 0.011903683205418056 2.259132956719706 0.008337580816090939 0.03536382246217887&
5 -0.006621737018652023 -0.004848655783835753 -0.006622894344592526 -0.026680695834797048 -0.009536352529412656&
6 0.06589011969329195 -0.02061858362766938 -0.01996275803193492 -0.005684409326651412 -0.008274484883510824&
7 -0.011570387361426513 -0.06063410167907909 0.013200239389132591 -0.03653817989122159 0.029218155517063922&
8 0.02710172178454872 -0.024774791113409026
9 #PF &
10 3.0 3.0 0.0 26.0 26.0&
11 1.8 1000.0 3.0 0.0 0.0&
12 2.1 2.2 2.3 2.4 2.5&
13 2.6 2.7 2.8 2.9 3.0&
14 3.1 3.2 3.3 3.4 3.5&
15 3.6 3.7 3.8 3.9 4.0&
16 4.4 4.6 4.8 5.2 5.6&
17 6.0 6.3

```

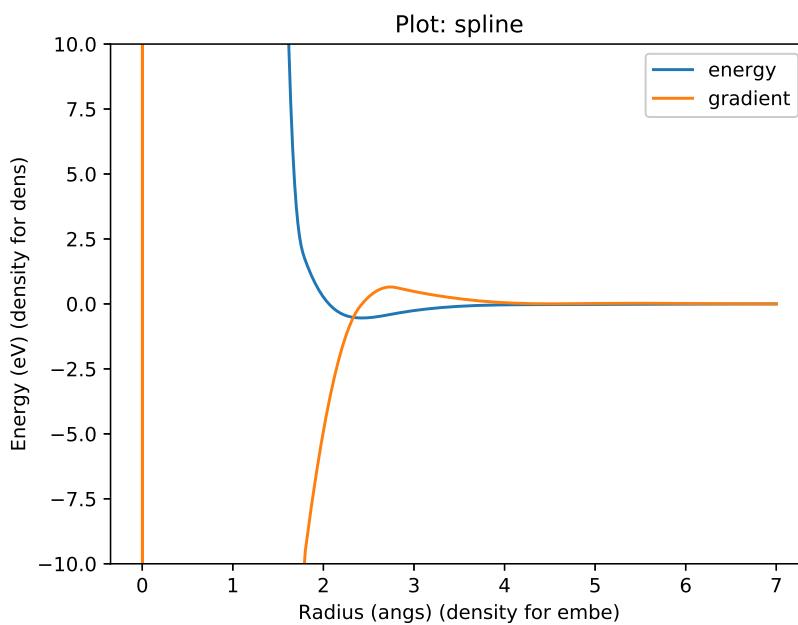


Figure T.7: Spline

T.2.8 Slater 4s

Listing T.15: Slater 4s Fortran code

```

1 !# Slater 4S
2
3 SUBROUTINE slater_4s(r, p, p_fixed, y)
4 !#####
5 ! f(x) = (N r^3 e^(-z r))^2
6 IMPLICIT NONE
7 !#####
8 REAL(kind=DoubleReal), INTENT(IN) :: r
9 REAL(kind=DoubleReal), INTENT(IN) :: p(:)
10 REAL(kind=DoubleReal), INTENT(IN) :: p_fixed(:)
11 REAL(kind=DoubleReal), INTENT(OUT) :: y
12 !#####
13 !#####
14 IF(SIZE(p) .EQ. 2)THEN
15   y = (p(1) * r**3 * exp(-1.0D0 * p(2) * r))**2

```

```

17 ELSE IF(SIZE(p) .EQ. 4)THEN
18   y = (p(1) * r**3 * exp(-1.0D0 * p(2) * r))**2 + &
19     (p(3) * r**3 * exp(-1.0D0 * p(4) * r))**2
20 END IF
21
22 IF(SIZE(p_fixed) .EQ. 1)THEN
23   IF(p_fixed(1) .GT. 0.0D0)THEN
24     CALL cutoff(r, p_fixed(1), 3, y)
25   END IF
26 END IF
27
28
29 END SUBROUTINE slater_4s
30
31
32 SUBROUTINE slater_4s_v(r, p, p_fixed, y)
33 !#####
34 ! f(x) = (N r^3 e^(-z r))^2
35 IMPLICIT NONE
36 !#####
37 REAL(kind=DoubleReal), INTENT(IN) :: r(:)
38 REAL(kind=DoubleReal), INTENT(IN) :: p(:)
39 REAL(kind=DoubleReal), INTENT(IN) :: p_fixed(:)
40 REAL(kind=DoubleReal), INTENT(OUT) :: y(1:SIZE(r,1))
41 INTEGER(kind=StandardInteger) :: n
42 !#####
43 DO n = 1, SIZE(r,1)
44   CALL slater_4s(r(n), p, p_fixed, y(n))
45 END DO
46 END SUBROUTINE slater_4s_v

```

Listing T.16: Slater 4s EAMPA input file

```

1 #TYPE slater_4s
2 #P    5.0 1.323
3 #PF    0.0

```

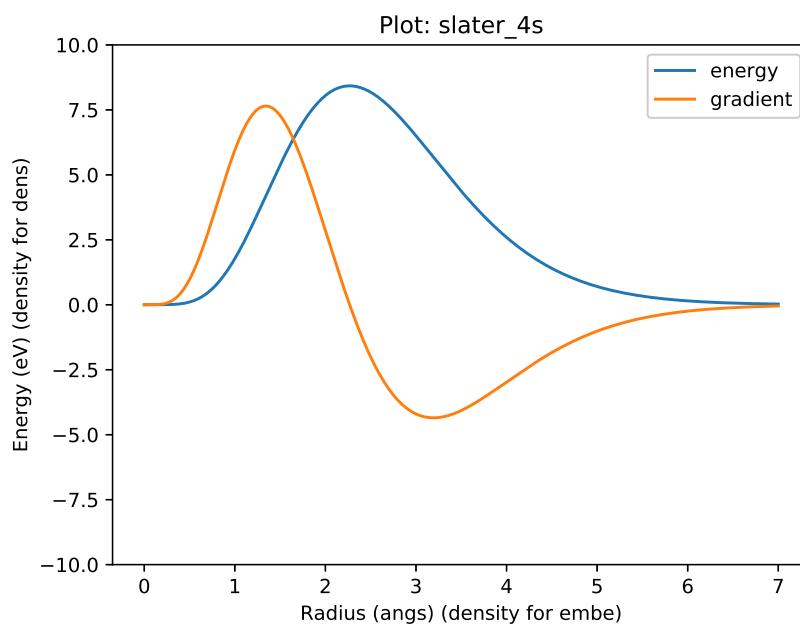


Figure T.8: Slater 4s

T.2.9 Zero

Listing T.17: Zero Fortran code

```

1 ! Just a zero function
2 ! f(x) = 0
3 ! Can be used to turn off certain potential functions
4
5 SUBROUTINE zero(r, p, p_fixed, y)
6 !#####
7 ! f(x) = 0
8 IMPLICIT NONE
9 !#####
10 REAL(kind=DoubleReal), INTENT(IN) :: r
11 REAL(kind=DoubleReal), INTENT(IN) :: p(:)
12 REAL(kind=DoubleReal), INTENT(IN) :: p_fixed(:)
13 REAL(kind=DoubleReal), INTENT(OUT) :: y
14 !#####
15 !#####
16 y = 0.0D0
17 END SUBROUTINE zero
18
19
20
21 SUBROUTINE zero_v(r, p, p_fixed, y)
22 !#####
23 ! f(x) = 0
24 IMPLICIT NONE
25 !#####
26 REAL(kind=DoubleReal), INTENT(IN) :: r(:)
27 REAL(kind=DoubleReal), INTENT(IN) :: p(:)
28 REAL(kind=DoubleReal), INTENT(IN) :: p_fixed(:)
29 REAL(kind=DoubleReal), INTENT(OUT) :: y(1:SIZE(r,1))
30 INTEGER(kind=StandardInteger) :: n
31 !#####
32 DO n = 1, SIZE(r,1)
33   CALL zero(r(n), p, p_fixed, y(n))
34 END DO
35 END SUBROUTINE zero_v

```

Listing T.18: Zero EAMPA input file

```

1 #TYPE  zero
2 #P     0.0
3 #PF    0.0

```

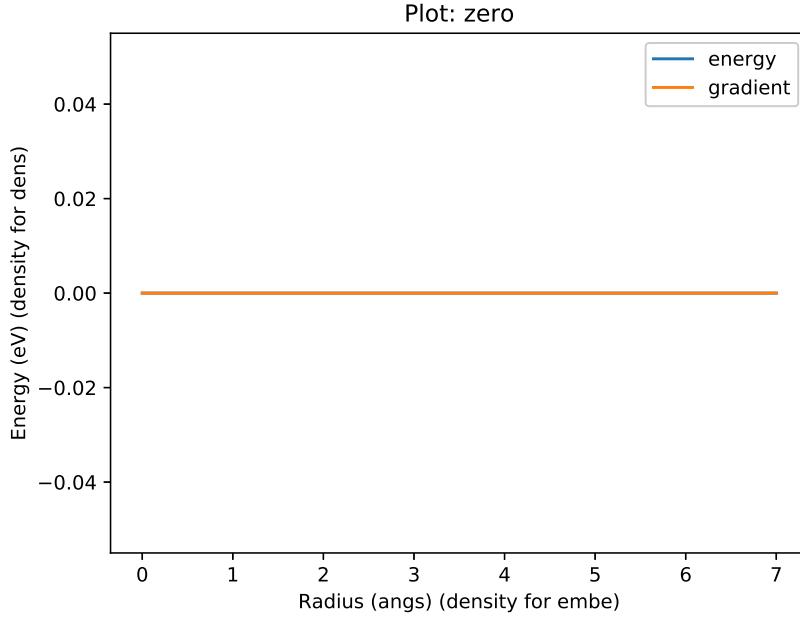


Figure T.9: Zero

T.3 Splines

Polynomial Splines

The form of polynomial spline used in the literature is a sum of N cubic polynomials that, by way of the Heaviside step function and their form, cutoff neatly at the desired radius. The cutoff radii are fixed and, during the fit of a potential, just the coefficients of each cubic spline are varied (eq. T.5).

$$V(r) = \sum_i^N a_i(r - r_i)^3 H(r_i - r) \quad \text{where} \quad (T.1)$$

$$H(x) = \begin{cases} 0 & x < 0 \\ 1 & x \geq 0 \end{cases}$$

If a continuous second derivative is required the cubic spline may be replaced with a quintic spline (eq. T.2).

$$V(r) = \sum_i^N a_i(r - r_i)^5 H(r_i - r) \quad (T.2)$$

Polynomial Knot to Knot Splines

So far, analytic potentials have been discussed. There are existing potentials that do not have an analytic form, but are tabulated data that have the properties that they are continuous and have a continuous first derivatives. In attempting to fit or re-fit tabulated data, it would be problematic to adjust each data point individually given

the number of data points and the requirement to have a continuous well behaved function with continuous first derivatives.

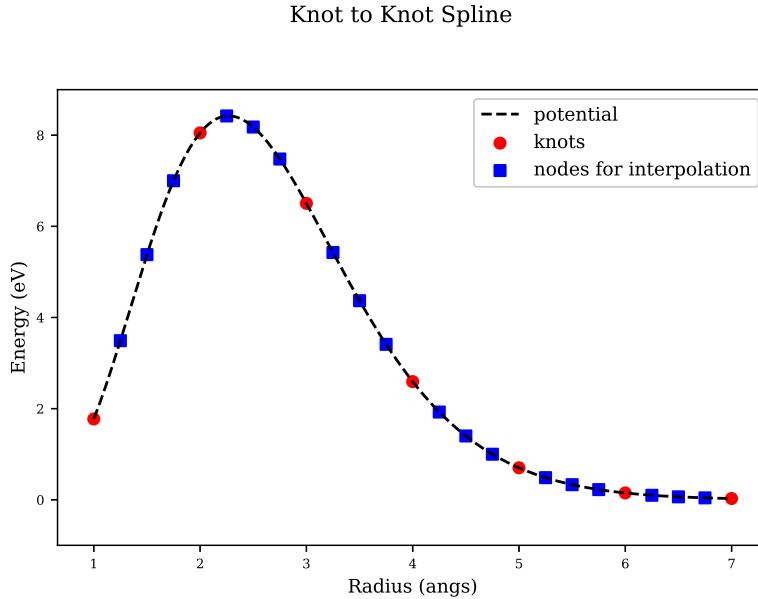


Figure T.10: Polynomial knot to knot spline

The tabulated function is divided into sections, and the start and end of each section forms the knot of the spline. A polynomial may be splined between these knots, as shown in figure T.10, but the order of polynomial will depend on the data available. Immediately, the x and y positions are known: $x_a, x_b, f(x_a), f(x_b)$. A first order polynomial has two unknowns, and so these unknowns may be calculated using linear algebra.

$$\begin{aligned} c_0 + c_1 x_a &= y_a \\ c_0 + c_1 x_b &= y_b \end{aligned} \tag{T.3}$$

One of the requirements for the potential is to have a continuous first derivative, and this requires knowledge of the first derivative at each knot. In the figure, the knots are shown as circles. By taking several points near to each knot, shown as squares, the derivative at each knot may be calculated by interpolation. This allows a third order polynomial to be used (eq. T.4).

$$\begin{aligned} P_1 &= (x_A, f(x_A)) \\ P_2 &= (x_A, f'(x_A)) \\ P_3 &= (x_B, f(x_B)) \\ P_4 &= (x_B, f'(x_B)) \end{aligned} \tag{T.4}$$

$$\begin{aligned} c_0 + c_1 x_a + c_2 x_a^2 + c_3 x_a^3 &= y_a \\ 0 + c_1 + 2c_2 x_a + 3c_3 x_a^2 &= y'_a \\ c_0 + c_1 x_b + c_2 x_b^2 + c_3 x_b^3 &= y_b \\ 0 + c_1 + 2c_2 x_b + 3c_3 x_b^2 &= y'_b \end{aligned} \tag{T.5}$$

Rewriting as matrices:

$$\begin{bmatrix} 1 & x_a & x_a^2 & x_a^3 \\ 0 & 1 & 2x_a & 3x_a^2 \\ 1 & x_b & x_b^2 & x_b^3 \\ 0 & 1 & 2x_b & 3x_b^2 \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{bmatrix} = \begin{bmatrix} f(x_a) \\ f'(x_a) \\ f(x_b) \\ f'(x_b) \end{bmatrix} \quad (\text{T.6})$$

It is important for the function and its derivative to be continuous. If it is also deemed necessary to have a continuous second derivative, a quintic spline may be used. By interpolating and calculating the second order derivative at each knot, a 5th order polynomial may be splined between pairs of knots.

This approach may be modified to spline functions of the form $\exp(p(r))$ between knots, and this is useful for splining between a ZBL or similar repulsive function at small r and a function of a different form at larger r.

where

$$V(r) = \begin{cases} ZBL(r, q_1, q_2) & r \leq r_a \\ \exp(a + br + cr^2 + dr^3) & r_a < r < r_b \\ v(r) & r_b < r < r_{cut} \\ 0 & r \geq r_{cut} \end{cases} \quad (\text{T.7})$$

To spline the exponential the equations are set up in a similar way, but this time the system of equations are non-linear.

$$\begin{aligned} \exp(c_0 + c_1 x_a + c_2 x_a^2 + c_3 x_a^3) &= y_a \\ (c_1 + 2c_2 x_a + 3c_3 x_a^2) \exp(c_0 + c_1 x_a + c_2 x_a^2 + c_3 x_a^3) &= y'_a \\ \exp(c_0 + c_1 x_b + c_2 x_b^2 + c_3 x_b^3) &= y_b \\ (c_1 + 2c_2 x_b + 3c_3 x_b^2) \exp(c_0 + c_1 x_b + c_2 x_b^2 + c_3 x_b^3) &= y'_b \end{aligned} \quad (\text{T.8})$$

Newton-Gauss is used to solve this problem where initial parameters are set for the constants in the equation and varying until the function values and derivatives at point A and point B match the values being fitted to.

Appendix U

Fe-Pd and Fe-Ru Structures

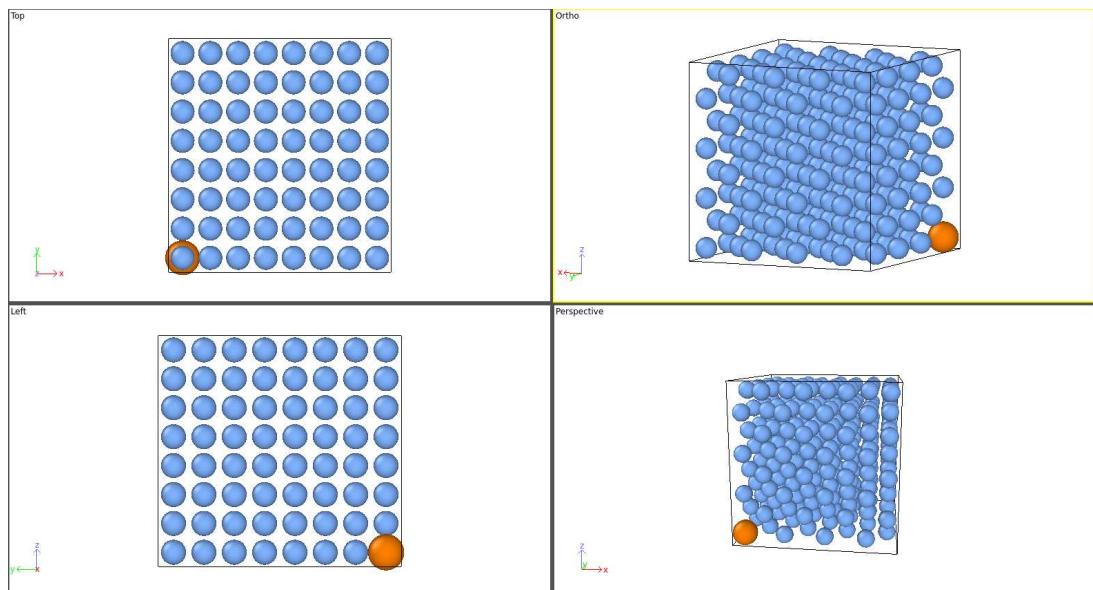


Figure U.1: FePd Structure 01

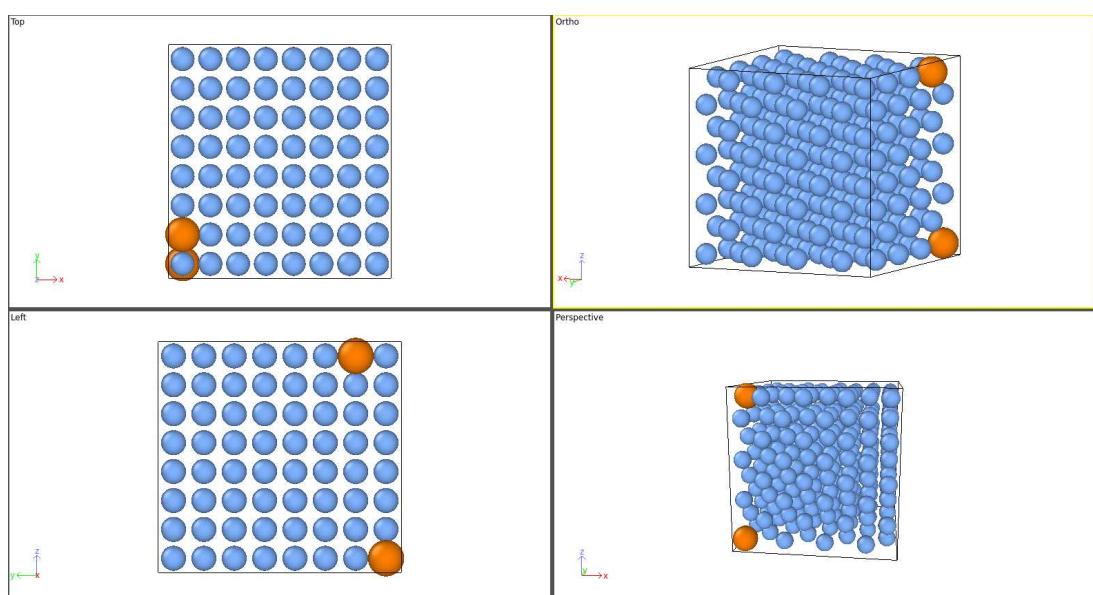


Figure U.2: FePd Structure 02

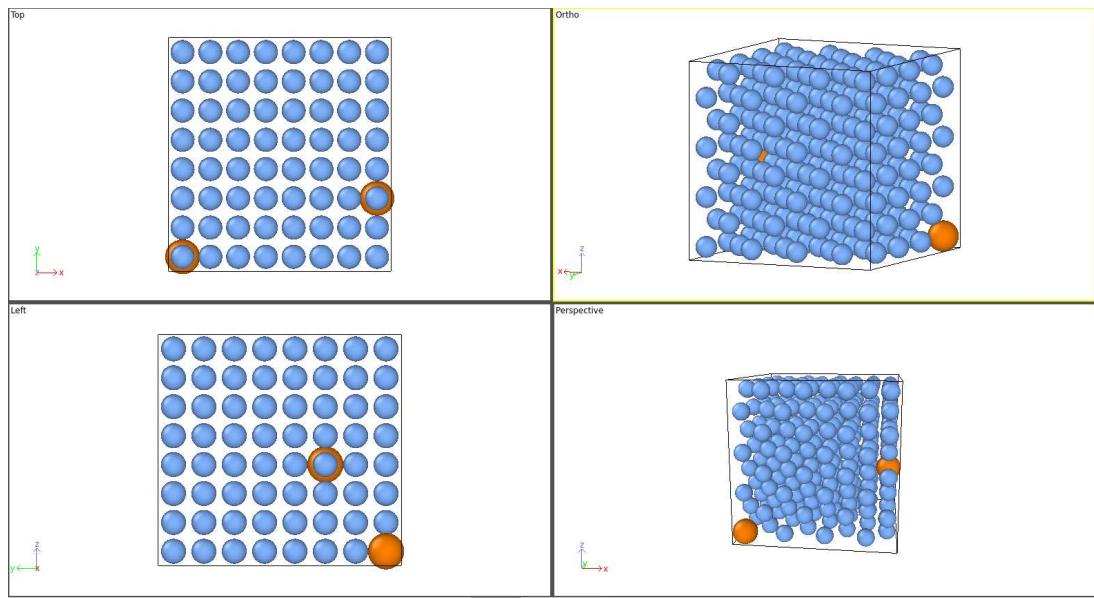


Figure U.3: FePd Structure 03

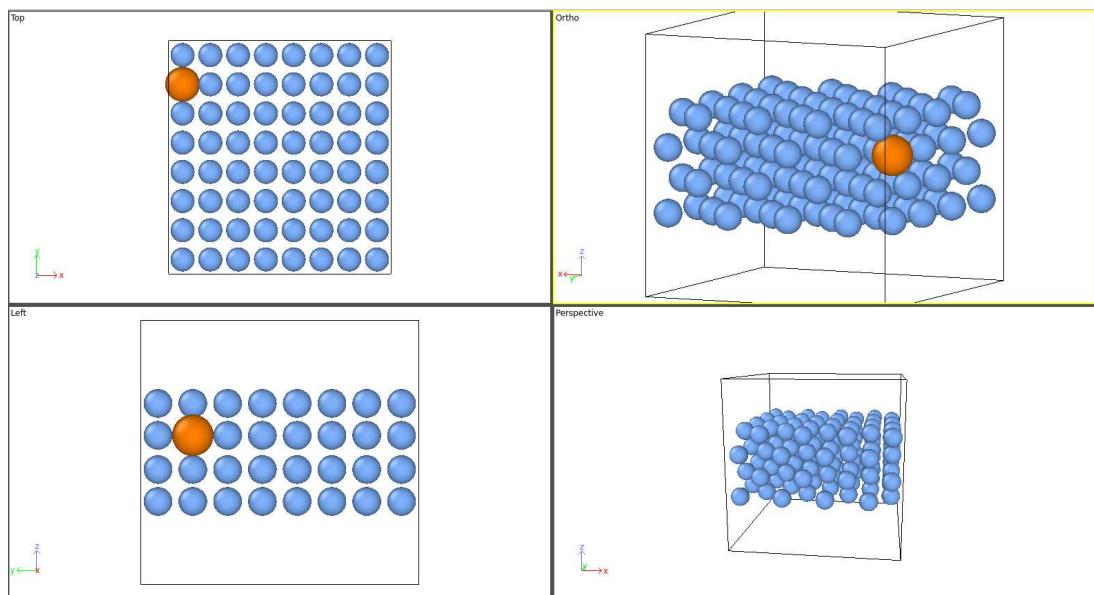


Figure U.4: FePd Structure 04

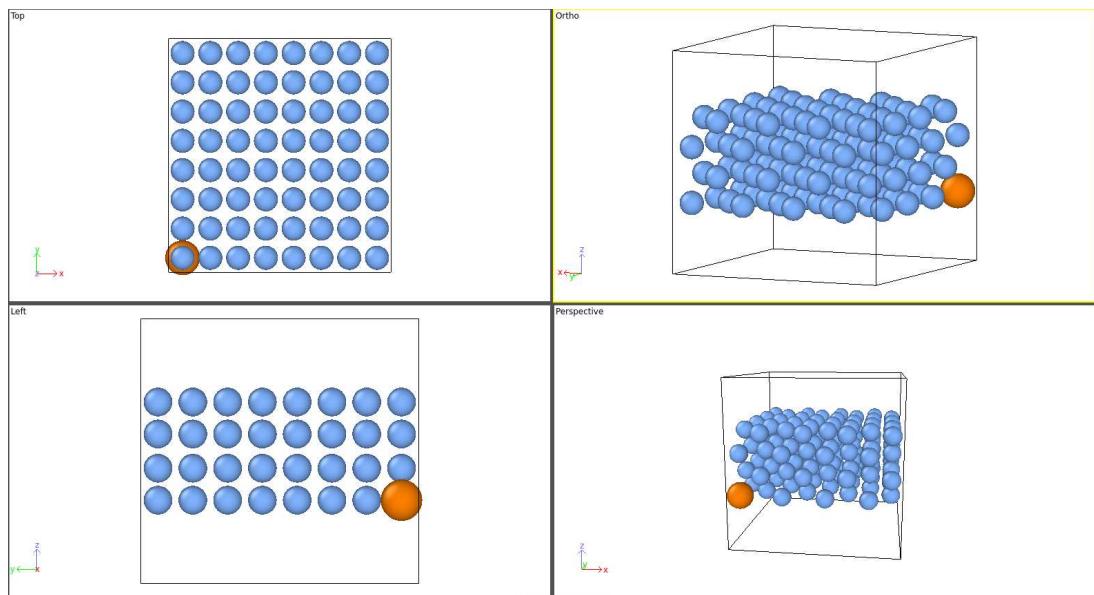


Figure U.5: FePd Structure 05

Appendix V

Fe-Pd Potential

This potential has been developed specifically for Fe and Pd where the atoms are in an FCC structure. Particular weighting was given to the relaxed configurations of each pure element, those with small additions of Pd to Fe and surface/slab configurations.

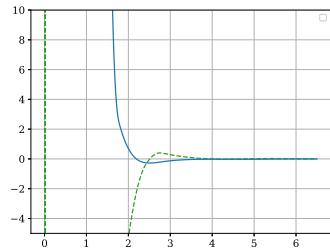
V.1 Tabulated Potential

The files for the potential are large, with the LAMMPS version being over 7,000 lines long containing 35,000+ data points. They are not included here but are available to download from:

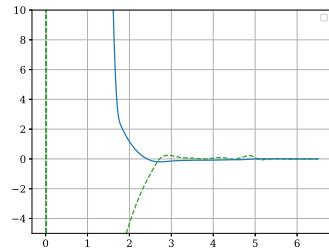
https://github.com/BenPalmer1983/fepd_feru_potential

V.2 Potential Function Plots

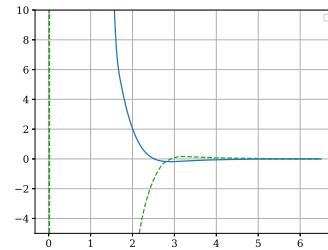
The seven functions that make up the potential are plotted in figure V.1.



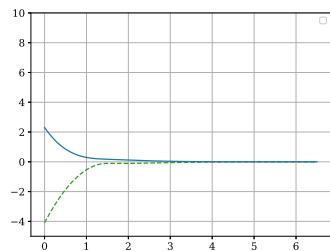
(a) Fe-Fe Pair



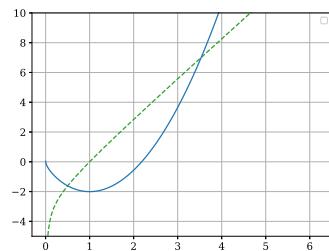
(b) Fe-Pd Pair



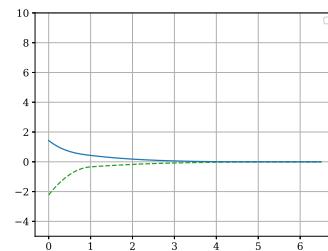
(c) Pd-Pd Pair



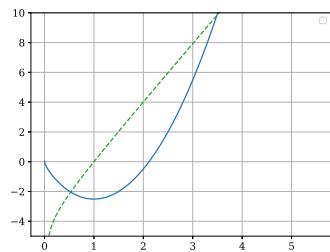
(d) Fe Density



(e) Fe Embed



(f) Pd Density



(g) Pd Embed

Figure V.1: Binary Alloy Potential for Fe-Pd

V.3 Potential vs DFT

The potentials were derived with a larger weight applied to a smaller set of configurations. How well the potential fits to the DFT results in terms of energy is given in figure V.2.

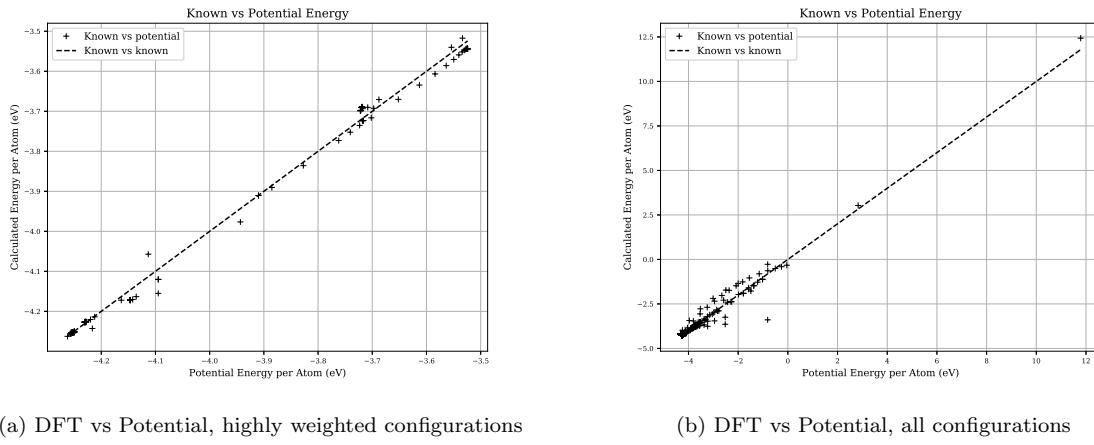


Figure V.2: Fe-Pd - a measure of how well the potential fits to first principles calculations

The DFT computed surface energy, as the slab forms from the increased gap between two surfaces, is given in figure V.3.

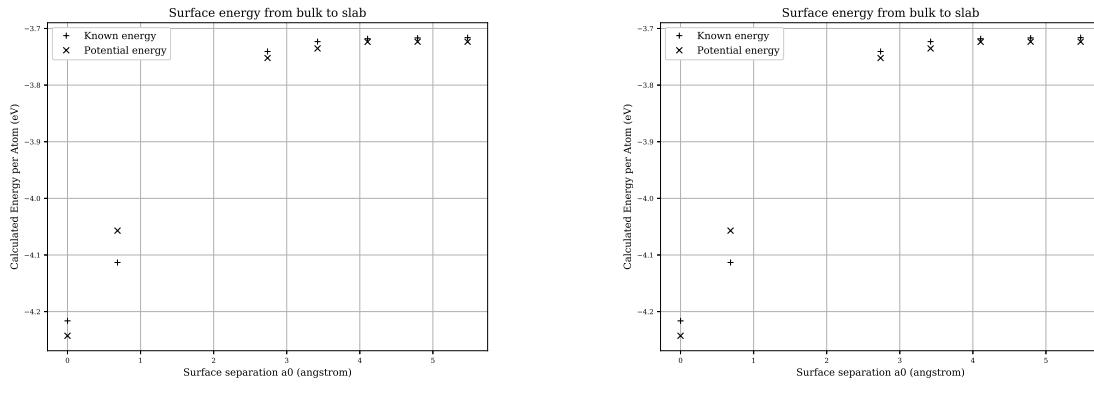


Figure V.3: Equation of State plots from this potential for FCC Fe

V.4 Bulk Properties

V.4.1 Iron FCC

Bulk Property Values

Property	DFT Computed	This Potential
a_0 (angs)	3.42	3.42
Basis	$\begin{bmatrix} 1.0 & 0.0 & 0.0 \\ 0.0 & 1.05 & 0.0 \\ 0.0 & 0.0 & 1.0 \end{bmatrix}$	$\begin{bmatrix} 1.0 & 0.0 & 0.0 \\ 0.0 & 1.05 & 0.0 \\ 0.0 & 0.0 & 1.0 \end{bmatrix}$
E_{coh} (eV)	-4.26	-4.27
B_0 (GPa)	226.1(a) 222.0(b)	246.6(a) 235.2(b)
Stiff.1 (GPa)	$\begin{bmatrix} 364.6 & 141.6 & 233.8 & 0 & 0 & 0 \\ 141.6 & 298.7 & 130.40 & 0 & 0 & 0 \\ 233.8 & 130.4 & 364.6 & 0 & 0 & 0 \\ 0 & 0 & 0 & 186.3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 266.8 & 0 \\ 0 & 0 & 0 & 0 & 0 & 186.3 \end{bmatrix}$	$\begin{bmatrix} 312.6 & 182.3 & 222.8 & 0 & 0 & 0 \\ 182.3 & 321.6 & 182.3 & 0 & 0 & 0 \\ 222.8 & 182.3 & 312.6 & 0 & 0 & 0 \\ 0 & 0 & 0 & 171.9 & 0 & 0 \\ 0 & 0 & 0 & 0 & 223.1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 171.9 \end{bmatrix}$
Stiff.2 (GPa)		$\begin{bmatrix} 329.3 & 205.3 & 205.3 & 0 & 0 & 0 \\ 205.3 & 329.3 & 205.3 & 0 & 0 & 0 \\ 205.3 & 205.3 & 329.3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 198.4 & 0 & 0 \\ 0 & 0 & 0 & 0 & 198.4 & 0 \\ 0 & 0 & 0 & 0 & 0 & 198.4 \end{bmatrix}$

Table V.1: The lattice parameter, cohesive energy and bulk modulus were determined by fitting the Birch-Murnaghan equation of state. The bulk modulus is computed from the Birch-Murnaghan equation of state (a) and the average Reuss and Voight Bulk Modulus (b). The elastic constants computed for the potential in stiffness (1) were computed using the method by Ravindran et al[130] and in stiffness (2) by Mehl et al[124][131].

Equation of State Plots

Both the Birch-Murnaghan and Rose-Vinet equation of states are used in the fitting process.

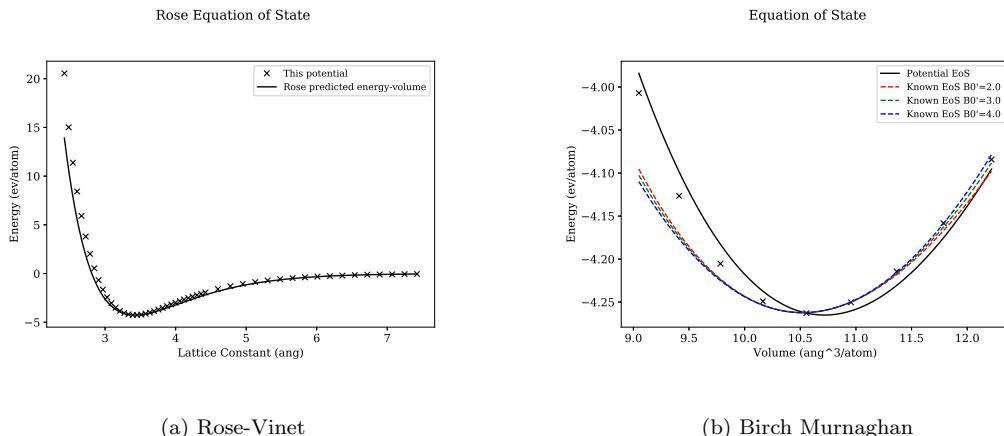


Figure V.4: Equation of State plots from this potential for FCC Fe

Elastic Constant Plots

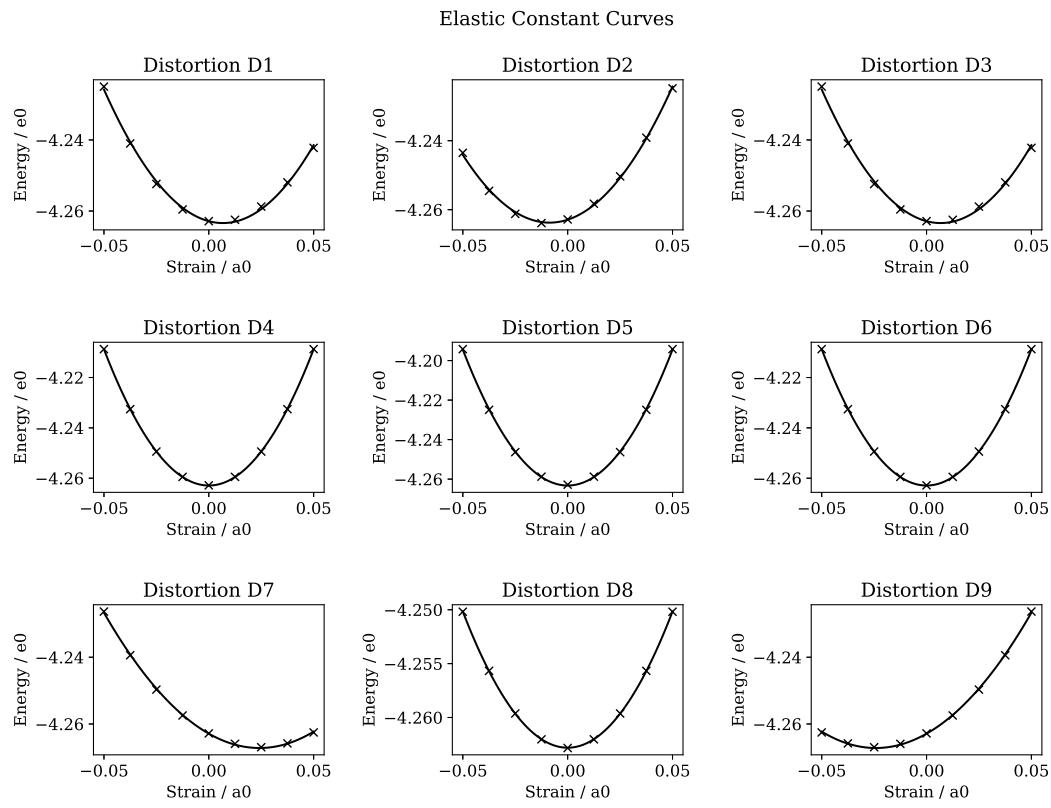
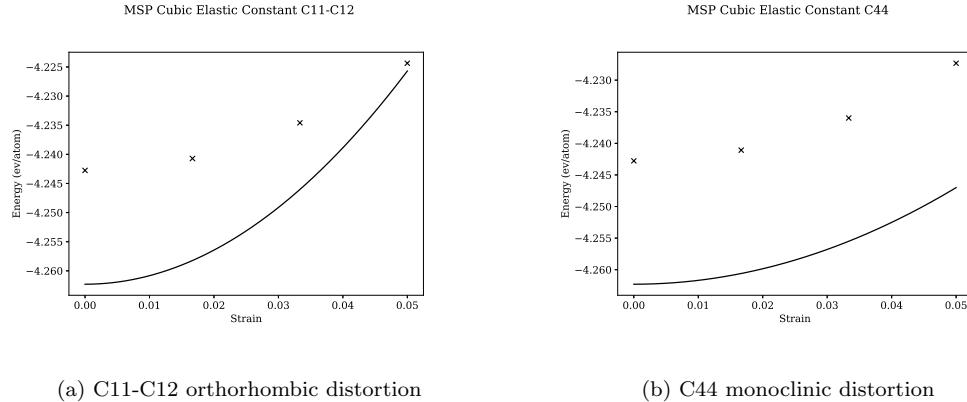


Figure V.5: Ravindran et al[130] distortion-energy plots for FCC Fe



(a) C11-C12 orthorhombic distortion

(b) C44 monoclinic distortion

Figure V.6: MSKP elastic constant strains, where the solid line is the known value and the points the values for this potential - FCC Fe

V.4.2 Palladium FCC

Property	DFT Computed	This Potential
a_0 (angs)	3.93	3.89
Basis	$\begin{bmatrix} 1.0 & 0.0 & 0.0 \\ 0.0 & 1.0 & 0.0 \\ 0.0 & 0.0 & 1.0 \end{bmatrix}$	$\begin{bmatrix} 1.0 & 0.0 & 0.0 \\ 0.0 & 1.0 & 0.0 \\ 0.0 & 0.0 & 1.0 \end{bmatrix}$
E_{coh} (eV)	-3.91	-3.78
B_0 (GPa)	184.4(a) 173.7(b)	165.0(a) 156.0(b)
Stiff.1 (GPa)	$\begin{bmatrix} 218.5 & 151.4 & 151.4 & 0 & 0 & 0 \\ 151.4 & 218.5 & 151.3 & 0 & 0 & 0 \\ 151.4 & 151.3 & 218.4 & 0 & 0 & 0 \\ 0 & 0 & 0 & 80.3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 80.3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 80.2 \end{bmatrix}$	$\begin{bmatrix} 187.1 & 140.3 & 140.3 & 0 & 0 & 0 \\ 140.3 & 187.1 & 140.3 & 0 & 0 & 0 \\ 140.3 & 140.3 & 187.1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 68.0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 68.0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 68.0 \end{bmatrix}$
Stiff.2 (GPa)		$\begin{bmatrix} 191.1 & 152.0 & 152.0 & 0 & 0 & 0 \\ 152.0 & 191.1 & 152.0 & 0 & 0 & 0 \\ 152.0 & 152.0 & 191.1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 58.8 & 0 & 0 \\ 0 & 0 & 0 & 0 & 58.8 & 0 \\ 0 & 0 & 0 & 0 & 0 & 58.8 \end{bmatrix}$

Table V.2: The lattice parameter, cohesive energy and bulk modulus were determined by fitting the Birch-Murnaghan equation of state. The bulk modulus is computed from the Birch-Murnaghan equation of state (a) and the average Reuss and Voight Bulk Modulus (b). The elastic constants computed for the potential in stiffness (1) were computed using the method by Ravindran et al[130] and in stiffness (2) by Mehl et al[124][131].

Equation of State Plots

Both the Birch-Murnaghan and Rose-Vinet equation of states are used in the fitting process.

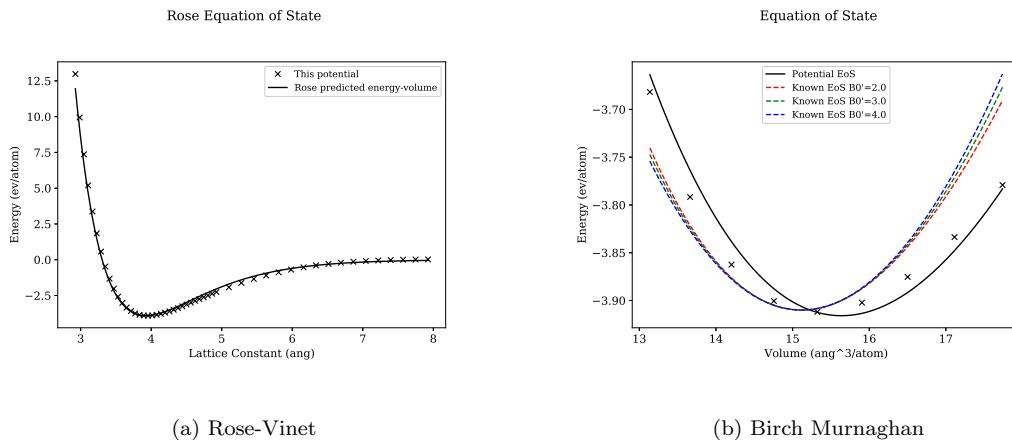


Figure V.7: Equation of State plots from this potential for FCC Pd

Elastic Constant Plots

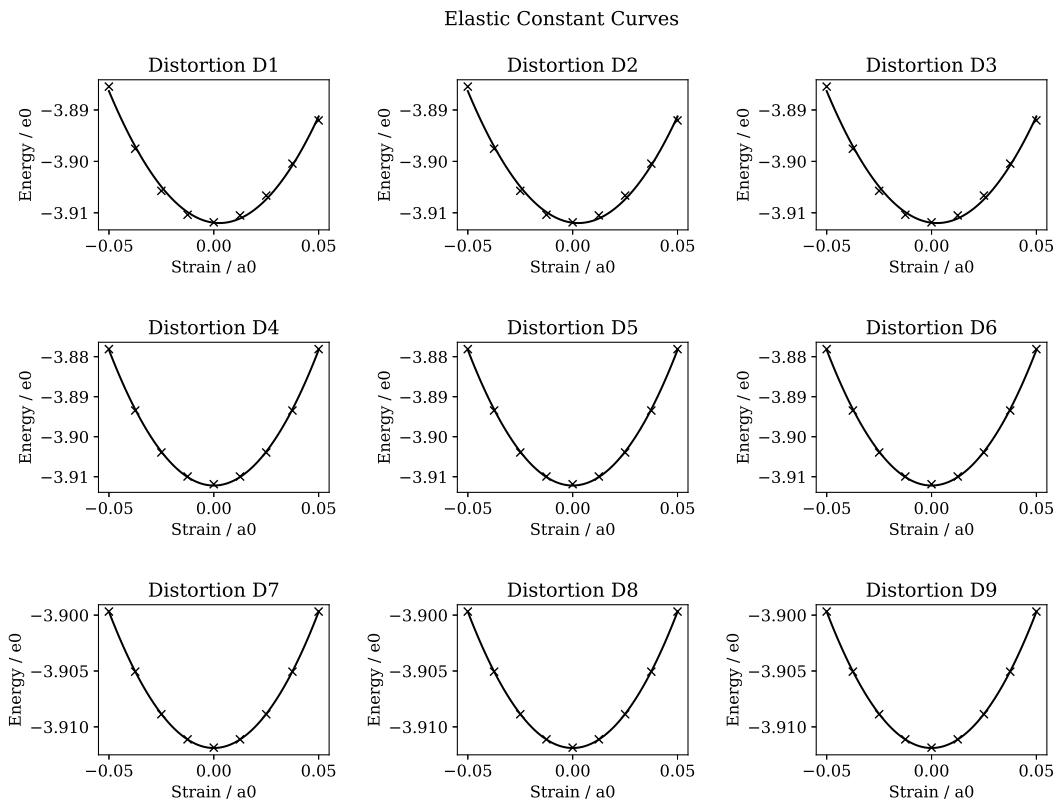
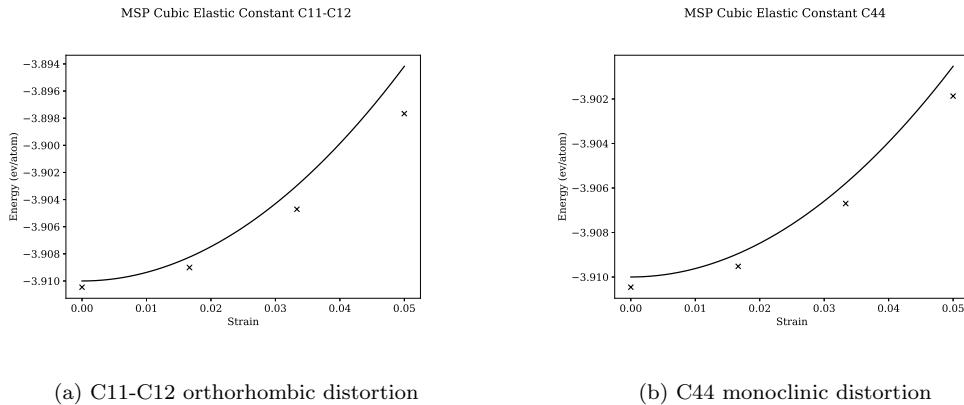


Figure V.8: Ravindran et al[130] distortion-energy plots for FCC Pd



(a) C11-C12 orthorhombic distortion

(b) C44 monoclinic distortion

Figure V.9: MSKP elastic constant strains, where the solid line is the known value and the points the values for this potential - FCC Pd

V.4.3 Iron BCC

Property	DFT Computed	This Potential
a_0 (angs)	2.80	2.76
Basis	$\begin{bmatrix} 1.0 & 0.0 & 0.0 \\ 0.0 & 1.0 & 0.0 \\ 0.0 & 0.0 & 1.0 \end{bmatrix}$	$\begin{bmatrix} 1.0 & 0.0 & 0.0 \\ 0.0 & 1.0 & 0.0 \\ 0.0 & 0.0 & 1.0 \end{bmatrix}$
E_{coh} (eV)	-4.32	-4.16
B_0 (GPa)	239.2(a) 205.3(b)	229.0(a) 207.6(b)
Stiff.1 (GPa)	$\begin{bmatrix} 250.1 & 182.3 & 183.3 & 0 & 0 & 0 \\ 182.3 & 249.1 & 182.9 & 0 & 0 & 0 \\ 183.2 & 182.9 & 251.2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 139.2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 139.7 & 0 \\ 0 & 0 & 0 & 0 & 0 & 139.3 \end{bmatrix}$	$\begin{bmatrix} 178.8 & 222.0 & 222.0 & 0 & 0 & 0 \\ 222.0 & 178.8 & 222.0 & 0 & 0 & 0 \\ 222.0 & 222.0 & 178.8 & 0 & 0 & 0 \\ 0 & 0 & 0 & 215.2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 215.2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 215.2 \end{bmatrix}$
Stiff.2 (GPa)		$\begin{bmatrix} 190.1 & 248.5 & 248.5 & 0 & 0 & 0 \\ 248.5 & 190.1 & 248.5 & 0 & 0 & 0 \\ 248.5 & 248.5 & 190.1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 168.7 & 0 & 0 \\ 0 & 0 & 0 & 0 & 168.7 & 0 \\ 0 & 0 & 0 & 0 & 0 & 168.7 \end{bmatrix}$

Table V.3: The lattice parameter, cohesive energy and bulk modulus were determined by fitting the Birch-Murnaghan equation of state. The bulk modulus is computed from the Birch-Murnaghan equation of state (a) and the average Reuss and Voight Bulk Modulus (b). The elastic constants computed for the potential in stiffness (1) were computed using the method by Ravindran et al[130] and in stiffness (2) by Mehl et al[124][131].

Equation of State Plots

Both the Birch-Murnaghan and Rose-Vinet equation of states are used in the fitting process.

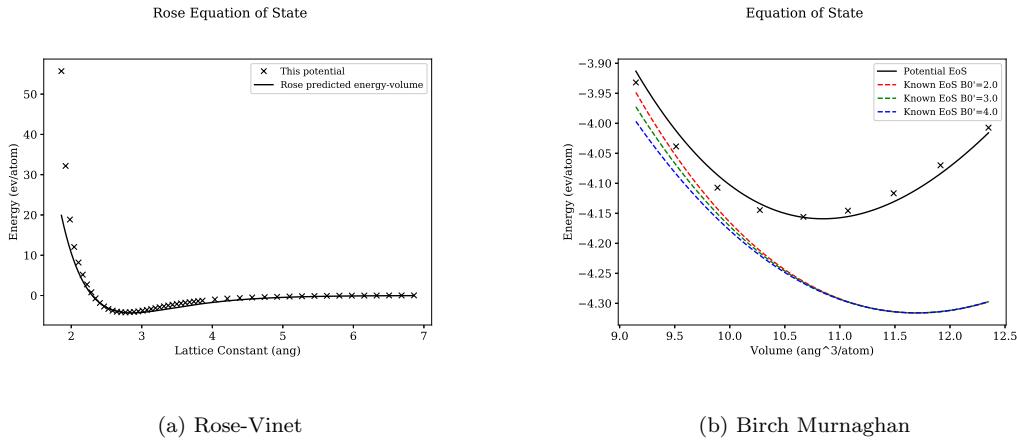


Figure V.10: Equation of State plots from this potential for FCC Pd

Elastic Constant Plots

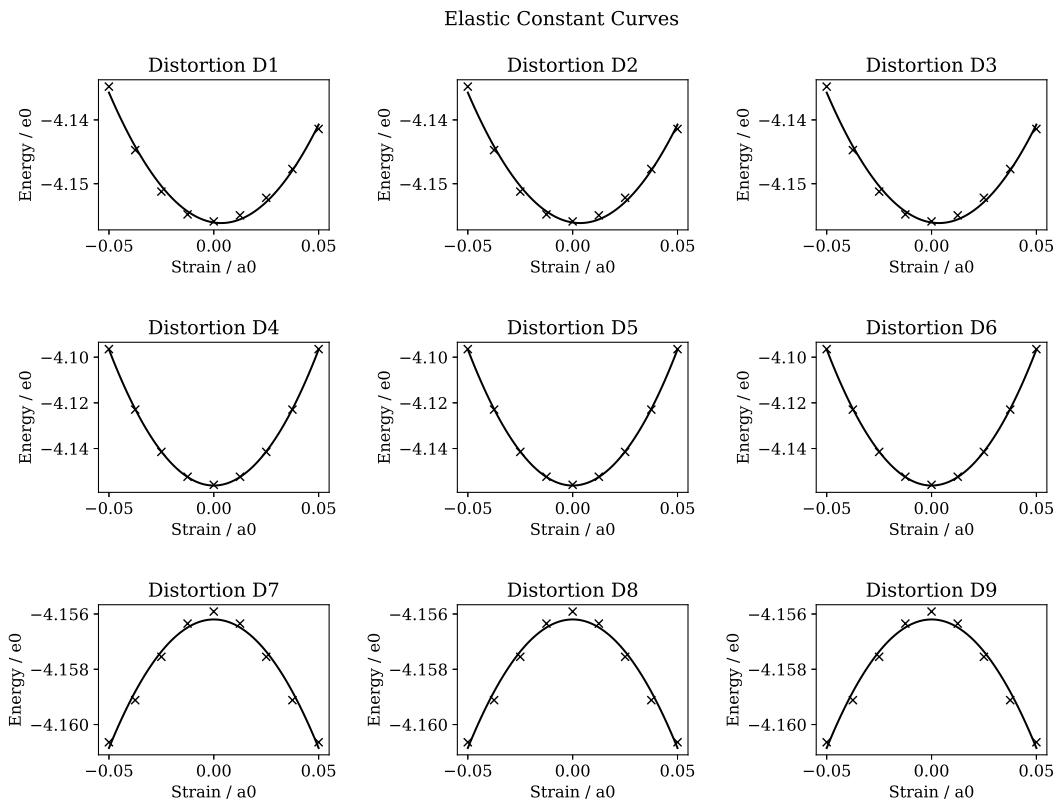


Figure V.11: Ravindran et al[130] distortion-energy plots for FCC Pd

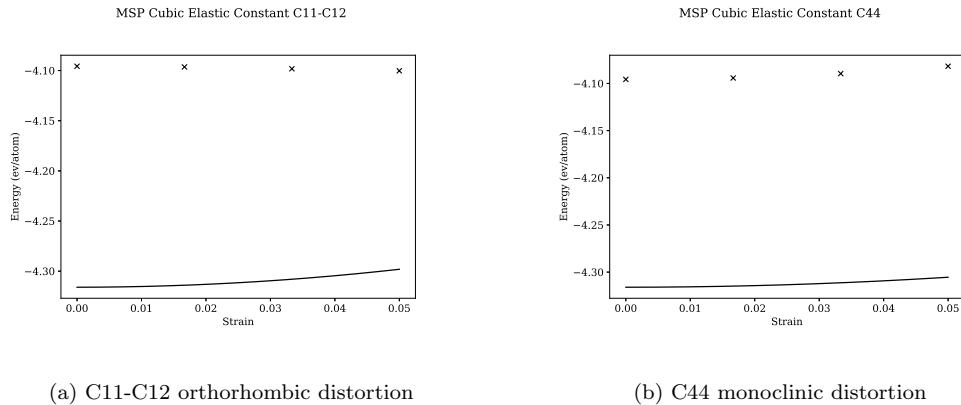


Figure V.12: MSKP elastic constant strains, where the solid line is the known value and the points the values for this potential - BCC Fe

Appendix W

Fe-Ru Potential

This potential has been developed specifically for Fe and Ru where the atoms are in an FCC structure. Particular weighting was given to the relaxed configurations of each pure element, those with small additions of Ru to Fe and surface/slab configurations.

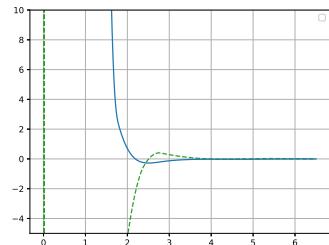
W.1 Tabulated Potential

The files for the potential are large, with the LAMMPS version being over 7,000 lines long containing 35,000+ data points. They are not included here but are available to download from:

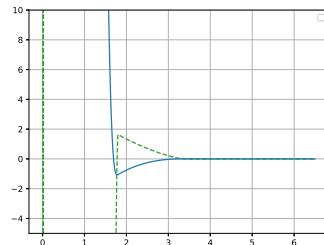
https://github.com/BenPalmer1983/fepd_feru_potential

W.2 Potential Function Plots

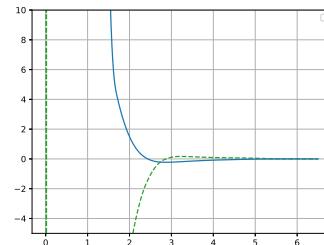
The seven functions that make up the potential are plotted in figure W.1.



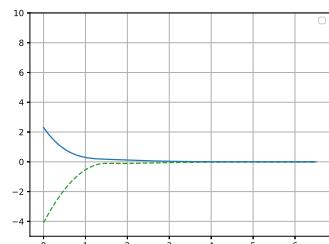
(a) Fe-Fe Pair



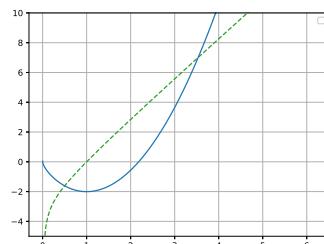
(b) Fe-Ru Pair



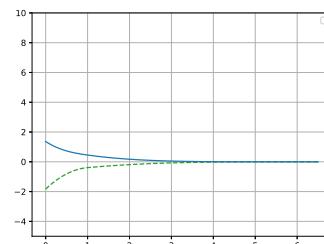
(c) Ru Ru Pair



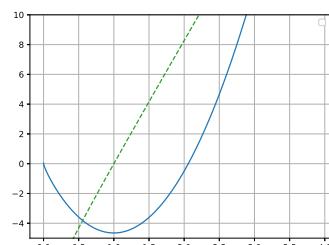
(d) Fe Density



(e) Fe Embed



(f) Ru Density

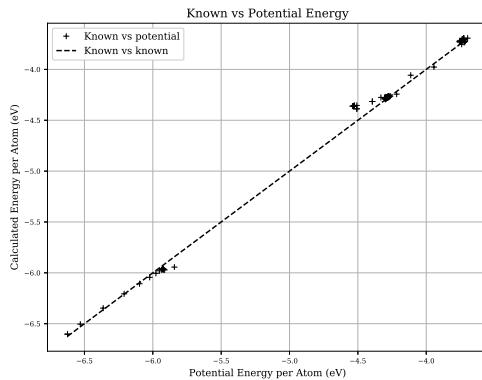


(g) Ru Embed

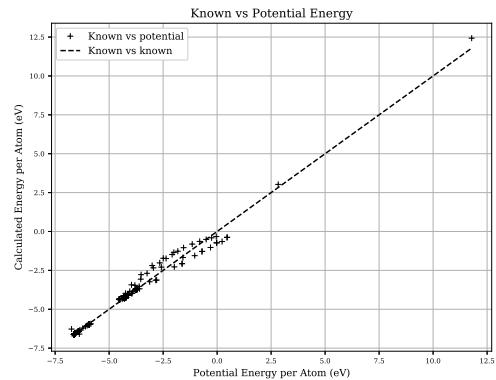
Figure W.1: Binary Alloy Potential for Fe-Ru

W.3 Potential vs DFT

The potentials were derived with a larger weight applied to a smaller set of configurations. How well the potential fits to the DFT results in terms of energy is given in figure W.2.



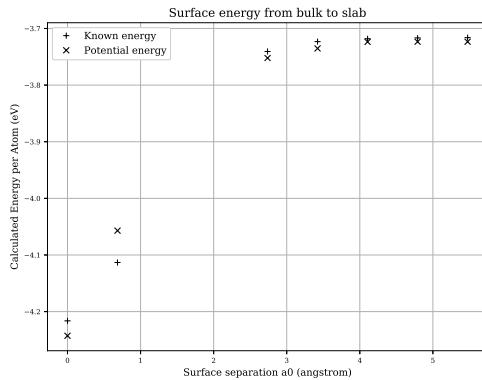
(a) DFT vs Potential, highly weighted configurations



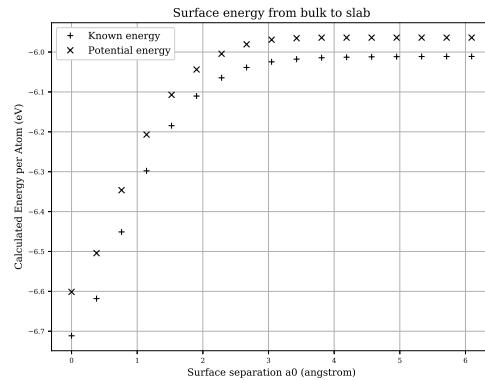
(b) DFT vs Potential, all configurations

Figure W.2: Fe-Ru - a measure of how well the potential fits to first principles calculations

The DFT computed surface energy, as the slab forms from the increased gap between two surfaces, is given in figure W.3.



(a) Fe surface energy at a range of surface separations



(b) Ru surface energy at a range of surface separations

Figure W.3: Equation of State plots from this potential for FCC Fe

W.4 Bulk Properties

W.4.1 Iron FCC

Bulk Property Values

Property	DFT Computed	This Potential
a_0 (angs)	3.42	3.42
Basis	$\begin{bmatrix} 1.0 & 0.0 & 0.0 \\ 0.0 & 1.05 & 0.0 \\ 0.0 & 0.0 & 1.0 \end{bmatrix}$	$\begin{bmatrix} 1.0 & 0.0 & 0.0 \\ 0.0 & 1.05 & 0.0 \\ 0.0 & 0.0 & 1.0 \end{bmatrix}$
E_{coh} (eV)	-4.26	-4.27
B_0 (GPa)	226.1(a) 222.0(b)	246.6(a) 235.2(b)
Stiff.1 (GPa)	$\begin{bmatrix} 364.6 & 141.6 & 233.8 & 0 & 0 & 0 \\ 141.6 & 298.7 & 130.40 & 0 & 0 & 0 \\ 233.8 & 130.4 & 364.6 & 0 & 0 & 0 \\ 0 & 0 & 0 & 186.3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 266.8 & 0 \\ 0 & 0 & 0 & 0 & 0 & 186.3 \end{bmatrix}$	$\begin{bmatrix} 312.6 & 182.3 & 222.8 & 0 & 0 & 0 \\ 182.3 & 321.6 & 182.3 & 0 & 0 & 0 \\ 222.8 & 182.3 & 312.6 & 0 & 0 & 0 \\ 0 & 0 & 0 & 171.9 & 0 & 0 \\ 0 & 0 & 0 & 0 & 223.1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 171.9 \end{bmatrix}$
Stiff.2 (GPa)		$\begin{bmatrix} 329.3 & 205.3 & 205.3 & 0 & 0 & 0 \\ 205.3 & 329.3 & 205.3 & 0 & 0 & 0 \\ 205.3 & 205.3 & 329.3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 198.4 & 0 & 0 \\ 0 & 0 & 0 & 0 & 198.4 & 0 \\ 0 & 0 & 0 & 0 & 0 & 198.4 \end{bmatrix}$

Table W.1: The lattice parameter, cohesive energy and bulk modulus were determined by fitting the Birch-Murnaghan equation of state. The bulk modulus is computed from the Birch-Murnaghan equation of state (a) and the average Reuss and Voight Bulk Modulus (b). The elastic constants computed for the potential in stiffness (1) were computed using the method by Ravindran et al[130] and in stiffness (2) by Mehl et al[124][131].

Equation of State Plots

Both the Birch-Murnaghan and Rose-Vinet equation of states are used in the fitting process.

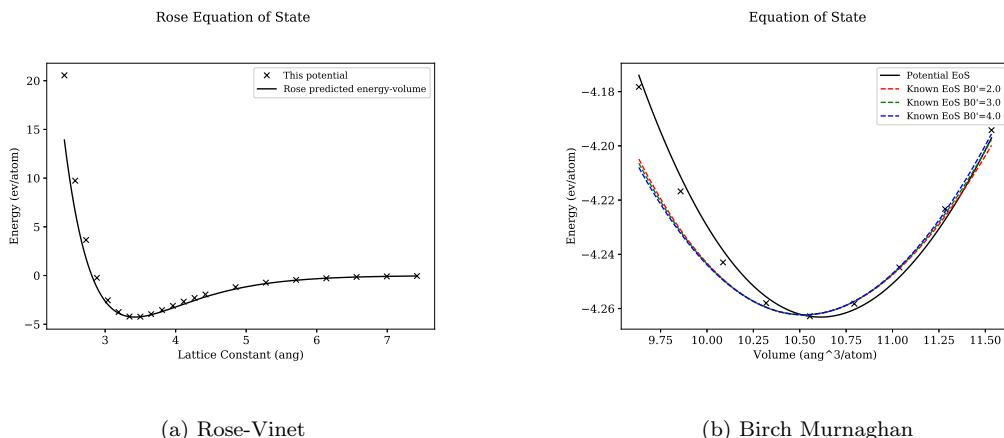


Figure W.4: Equation of State plots from this potential for FCC Fe

Elastic Constant Plots

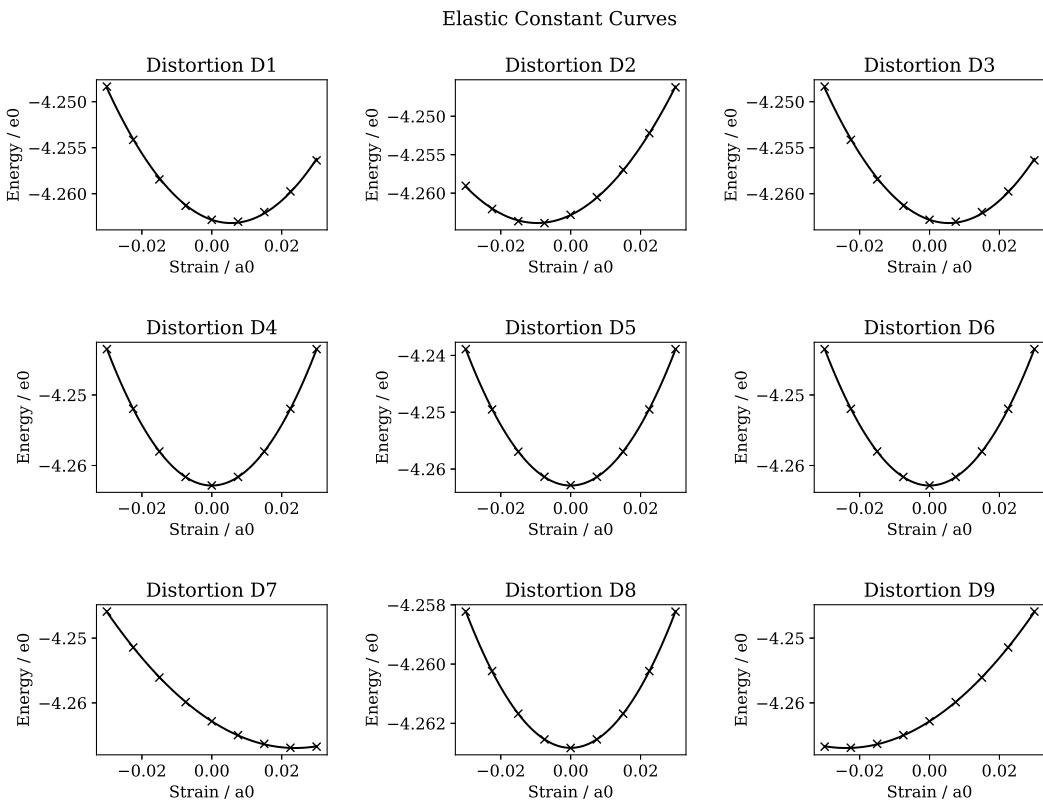


Figure W.5: Ravindran et al[130] distortion-energy plots for FCC Fe

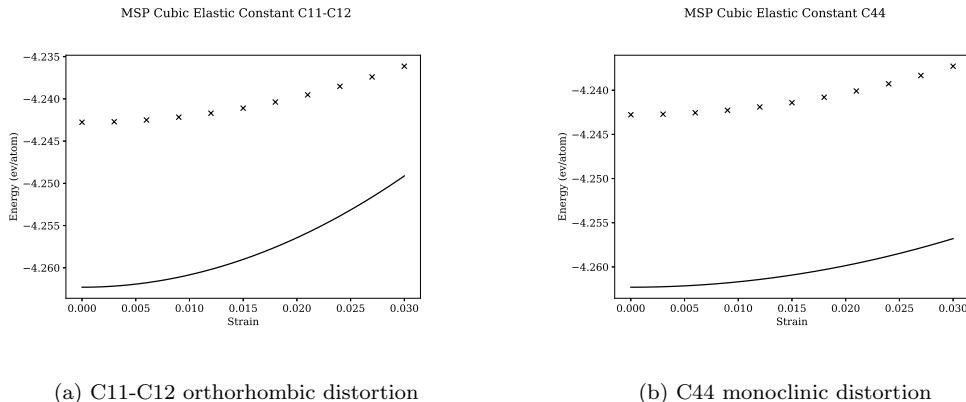


Figure W.6: MSKP elastic constant strains, where the solid line is the known value and the points the values for this potential - FCC Fe

W.4.2 Ruthenium FCC

Property	DFT Computed	This Potential
a_0 (angs)	3.81	3.79
Basis	$\begin{bmatrix} 1.0 & 0.0 & 0.0 \\ 0.0 & 1.0 & 0.0 \\ 0.0 & 0.0 & 1.0 \end{bmatrix}$	$\begin{bmatrix} 1.0 & 0.0 & 0.0 \\ 0.0 & 1.0 & 0.0 \\ 0.0 & 0.0 & 1.0 \end{bmatrix}$
E_{coh} (eV)	-6.62	-6.61
B_0 (GPa)	307.8(a) 303.3(b)	331.1(a) 320.4(b)
Stiff.1 (GPa)	$\begin{bmatrix} 471.5 & 219.1 & 219.2 & 0 & 0 & 0 \\ 219.1 & 471.5 & 219.2 & 0 & 0 & 0 \\ 219.2 & 219.2 & 471.7 & 0 & 0 & 0 \\ 0 & 0 & 0 & 245.0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 245.0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 245.0 \end{bmatrix}$	$\begin{bmatrix} 368.3 & 296.5 & 296.5 & 0 & 0 & 0 \\ 296.5 & 368.3 & 296.5 & 0 & 0 & 0 \\ 296.5 & 296.5 & 368.3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 104.0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 104.0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 104.0 \end{bmatrix}$
Stiff.2 (GPa)		$\begin{bmatrix} 375.0 & 309.2 & 309.2 & 0 & 0 & 0 \\ 309.2 & 375.0 & 309.2 & 0 & 0 & 0 \\ 309.2 & 309.2 & 375.0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 97.5 & 0 & 0 \\ 0 & 0 & 0 & 0 & 97.5 & 0 \\ 0 & 0 & 0 & 0 & 0 & 97.5 \end{bmatrix}$

Table W.2: The lattice parameter, cohesive energy and bulk modulus were determined by fitting the Birch-Murnaghan equation of state. The bulk modulus is computed from the Birch-Murnaghan equation of state (a) and the average Reuss and Voight Bulk Modulus (b). The elastic constants computed for the potential in stiffness (1) were computed using the method by Ravindran et al[130] and in stiffness (2) by Mehl et al[124][131].

Equation of State Plots

Both the Birch-Murnaghan and Rose-Vinet equation of states are used in the fitting process.

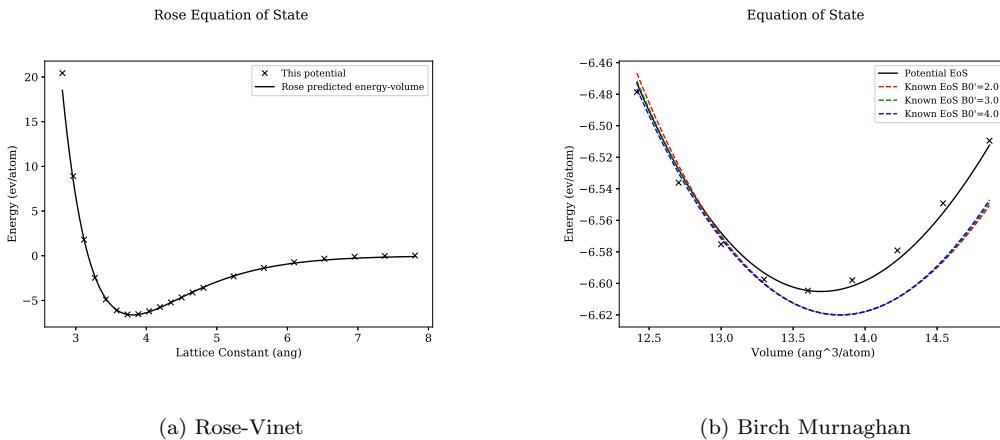


Figure W.7: Equation of State plots from this potential for FCC Ru

Elastic Constant Plots

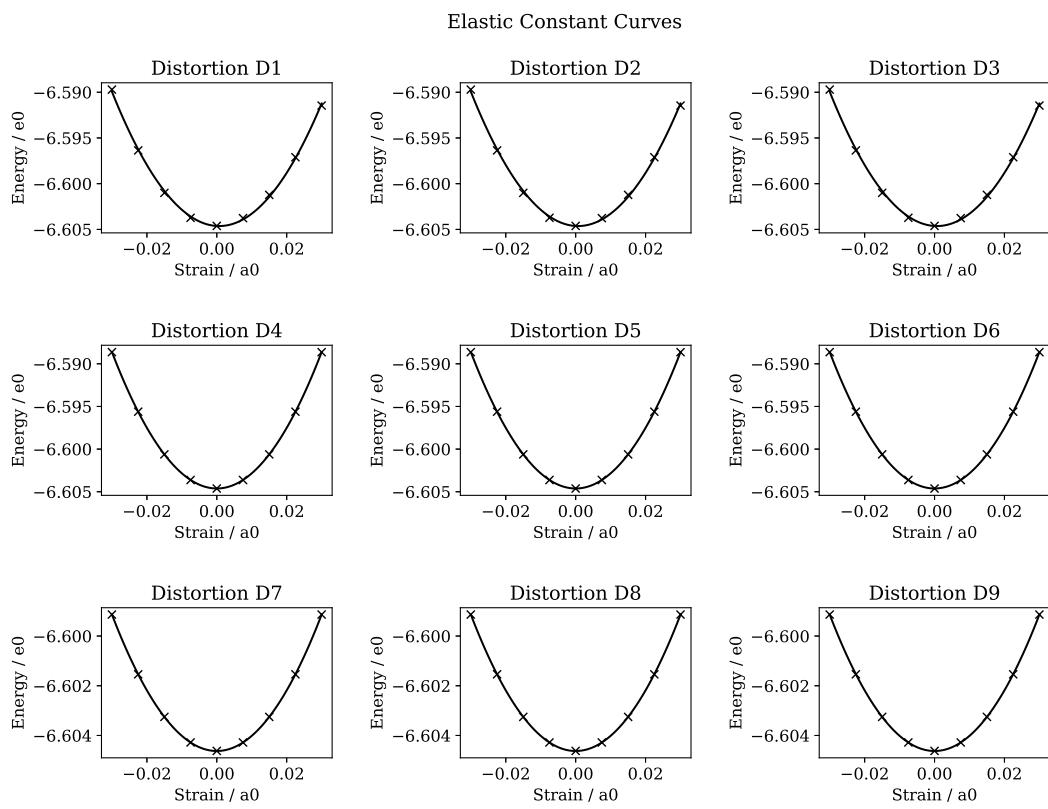
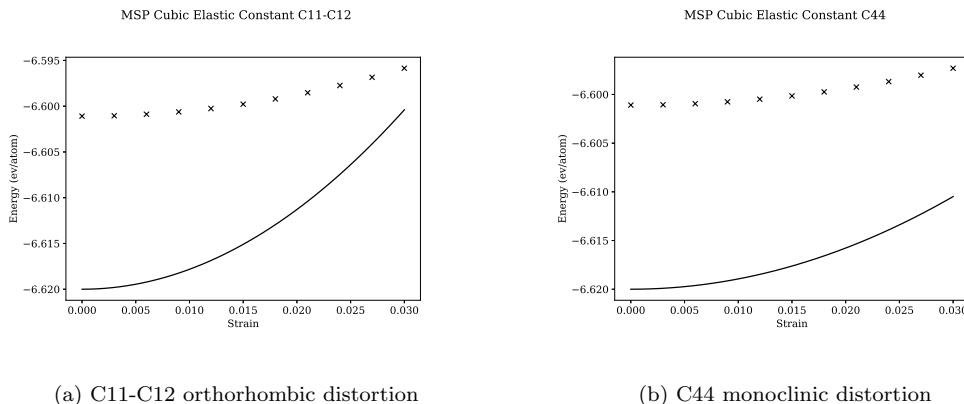


Figure W.8: Ravindran et al[130] distortion-energy plots for FCC Ru



(a) C11-C12 orthorhombic distortion

(b) C44 monoclinic distortion

Figure W.9: MSKP elastic constant strains, where the solid line is the known value and the points the values for this potential - FCC Ru

W.4.3 Iron BCC

Property	DFT Computed	This Potential
a_0 (angs)	2.80	2.76
Basis	$\begin{bmatrix} 1.0 & 0.0 & 0.0 \\ 0.0 & 1.0 & 0.0 \\ 0.0 & 0.0 & 1.0 \end{bmatrix}$	$\begin{bmatrix} 1.0 & 0.0 & 0.0 \\ 0.0 & 1.0 & 0.0 \\ 0.0 & 0.0 & 1.0 \end{bmatrix}$
E_{coh} (eV)	-4.32	-4.16
B_0 (GPa)	239.2(a) 205.3(b)	229.0(a) 207.6(b)
Stiff.1 (GPa)	$\begin{bmatrix} 250.1 & 182.3 & 183.3 & 0 & 0 & 0 \\ 182.3 & 249.1 & 182.9 & 0 & 0 & 0 \\ 183.2 & 182.9 & 251.2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 139.2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 139.7 & 0 \\ 0 & 0 & 0 & 0 & 0 & 139.3 \end{bmatrix}$	$\begin{bmatrix} 178.8 & 222.0 & 222.0 & 0 & 0 & 0 \\ 222.0 & 178.8 & 222.0 & 0 & 0 & 0 \\ 222.0 & 222.0 & 178.8 & 0 & 0 & 0 \\ 0 & 0 & 0 & 215.2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 215.2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 215.2 \end{bmatrix}$
Stiff.2 (GPa)		$\begin{bmatrix} 190.1 & 248.5 & 248.5 & 0 & 0 & 0 \\ 248.5 & 190.1 & 248.5 & 0 & 0 & 0 \\ 248.5 & 248.5 & 190.1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 168.7 & 0 & 0 \\ 0 & 0 & 0 & 0 & 168.7 & 0 \\ 0 & 0 & 0 & 0 & 0 & 168.7 \end{bmatrix}$

Table W.3: The lattice parameter, cohesive energy and bulk modulus were determined by fitting the Birch-Murnaghan equation of state. The bulk modulus is computed from the Birch-Murnaghan equation of state (a) and the average Reuss and Voight Bulk Modulus (b). The elastic constants computed for the potential in stiffness (1) were computed using the method by Ravindran et al[130] and in stiffness (2) by Mehl et al[124][131].

Equation of State Plots

Both the Birch-Murnaghan and Rose-Vinet equation of states are used in the fitting process.

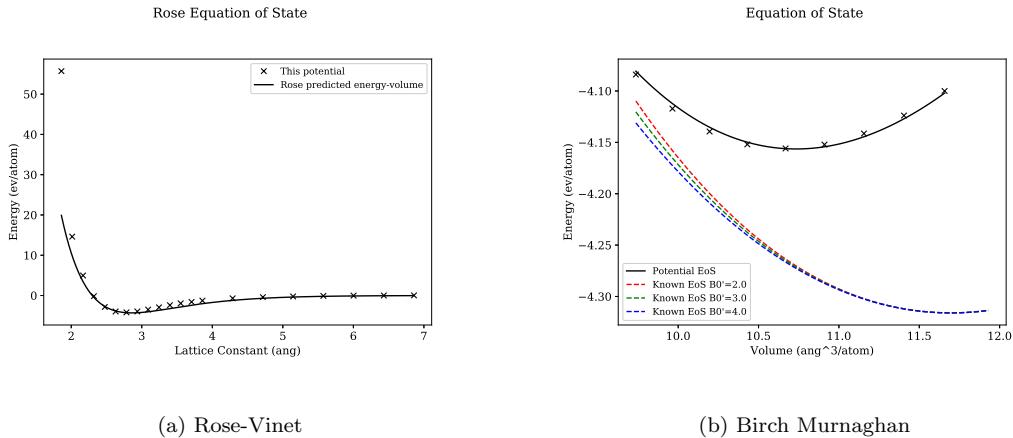


Figure W.10: Equation of State plots from this potential for BCC Fe

Elastic Constant Plots

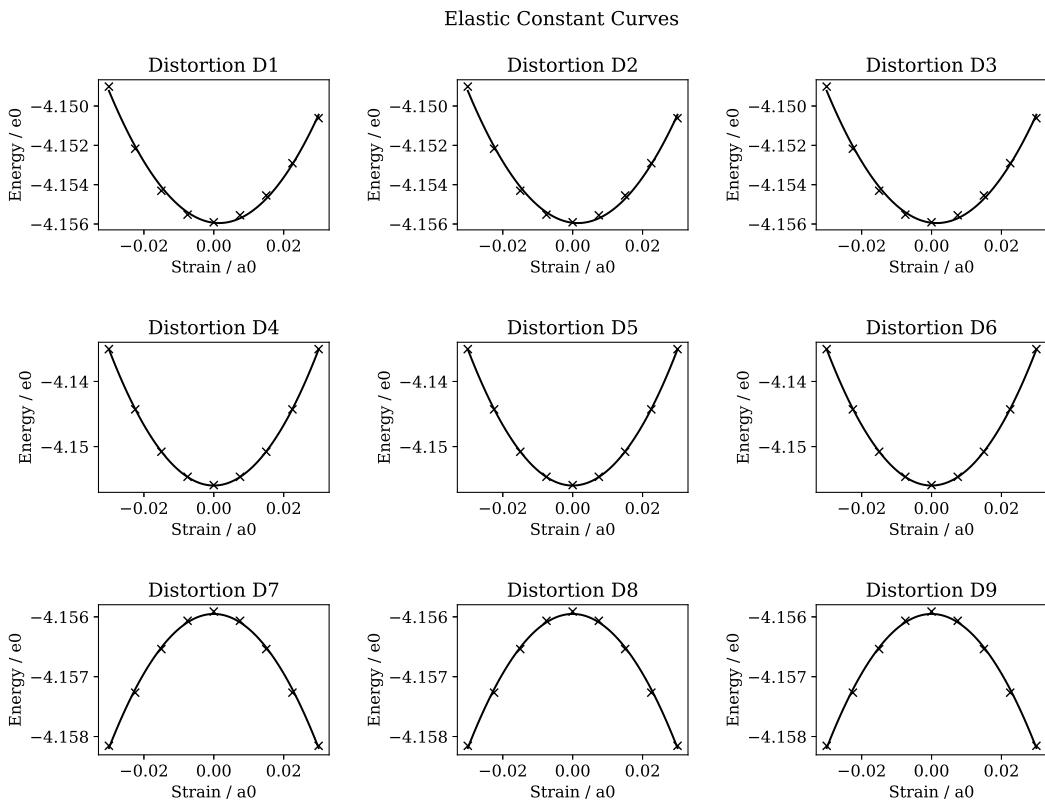
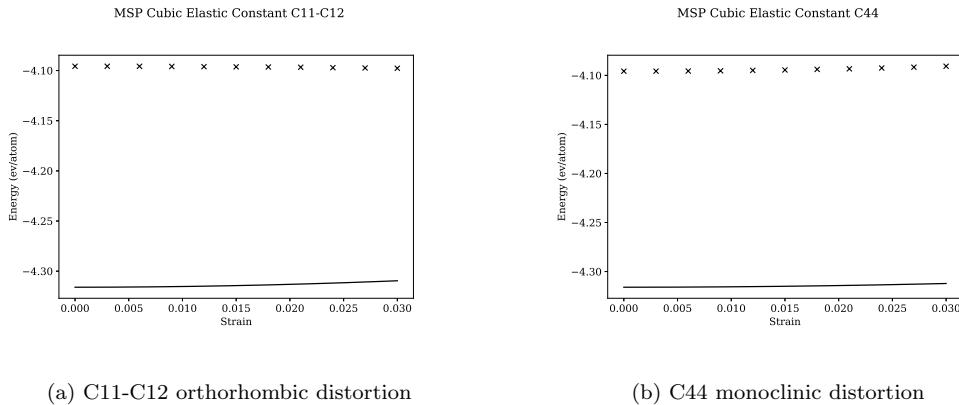


Figure W.11: Ravindran et al[130] distortion-energy plots for FCC Ru



(a) C11-C12 orthorhombic distortion

(b) C44 monoclinic distortion

Figure W.12: MSKP elastic constant strains, where the solid line is the known value and the points the values for this potential - BCC Ru

Appendix X

Contribution to the DL POLY Source Code

X.1 Contribution to DL POLY 4.04

I originally downloaded the 4.03 version of DL POLY while conducting my literature review. As a result of reading work on 2BEAM I made modifications to the Fortran code, specifically changing the arrays used to hold the potentials and retrieve the energy and forces from this modified metal potential. Dr. Todorov assisted me and adjusted my modifications such that the code worked across domains when running DL POLY over multiple processes.

Listing X.1: Add two numbers function

```
1 NEW FEATURES & IMPROVEMENTS
2 -----
3
4 1. New two band (2B) EAM and EEAM potentials for metals (TEST45 and
5 TEST46).
6
7 Acknowledgements
8 -----
9 Ben Palmer @ University of Birmingham (UK) for contributing to the
10 development and testing of the 2BEAM for metals;
```

Listing X.2: Add two numbers function

```
1 Subroutine metal_table_read(l_top)
2
3 !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
4 !
5 ! dl_poly_4 subroutine for reading potential energy and force arrays
6 ! from TABEAM file (for metal EAM & EEAM forces only)
7 !
8 ! copyright - daresbury laboratory
9 ! author - w.smith march 2006
10 ! amended - i.t.todorov march 2016
11 ! contrib - r.davidchak (eeam) june 2012
12 ! contrib - b.palmer (2band) may 2013
13 !
14 !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
15
16 Use kinds_f90
17 Use comms_module, Only : idnode,mxnnode,gsum
```

```

18 Use setup_module, Only : ntable,nrite,mxgmet,engunit
19 Use site_module, Only : ntpatm,unqatm
20 Use metal_module, Only : ntpmet,tabmet,lstmet,vmet,dmet,dmes,fmet,fmes
21 Use parse_module, Only : get_line,get_word,lower_case,word_2_real
22
23 Implicit None
24
25 Logical, Intent( In ) :: l_top
26
27 Logical :: safe
28 Character( Len = 200 ) :: record
29 Character( Len = 40 ) :: word
30 Character( Len = 4 ) :: keyword
31 Character( Len = 8 ) :: atom1,atom2
32 Integer :: fail(1:2),i,j,ipot,numpot,ktype,ngrid, &
33                      cp,cd,cds,ce,ces,katom1,katom2,keymet,k0,jtpatm
34 Real( Kind = wp ) :: start,finish
35
36 Integer,      Dimension( : ), Allocatable :: cpair, cdens,cdnss, &
37                                     cembed,cembds
38 Real( Kind = wp ), Dimension( : ), Allocatable :: buffer
39
40 fail=0
41 If      (tabmet == 1) Then ! EAM
42     Allocate (cpair(1:(ntpmet*(ntpmet+1))/2),cdens(1:ntpmet),
43                           cembed(1:ntpmet),                               &
44                           Stat=fail(1))
45 Else If (tabmet == 2) Then ! EEAM
46     Allocate (cpair(1:(ntpmet*(ntpmet+1))/2),cdens(1:ntpmet**2),
47                           cembed(1:ntpmet),                               &
48                           Stat=fail(1))
49 Else If (tabmet == 3) Then ! 2BEAM
50     Allocate (cpair(1:(ntpmet*(ntpmet+1))/2),cdens(1:ntpmet),cdnss(1:ntpmet*(ntpmet+1)/2),
51                           cembed(1:ntpmet),cembds(1:ntpmet), Stat=fail(1))
52 Else If (tabmet == 4) Then ! 2BEEAM
53     Allocate (cpair(1:(ntpmet*(ntpmet+1))/2),cdens(1:ntpmet**2),cdnss(1:ntpmet**2), &
54                           cembed(1:ntpmet),cembds(1:ntpmet), Stat=fail(1))
55 End If
56 Allocate (buffer(1:mxgmet),                               Stat=fail(2))
57 If (Any(fail > 0)) Then
58     Write(nrite,'(/,1x,a,i0)') 'metal_table_read allocation failure, node: ', idnode
59     Call error(0)
60 End If
61 cpair=0 ; cp=0
62 cdens=0 ; cd=0
63 cembed=0 ; ce=0
64 If (tabmet == 3 .or. tabmet == 4) Then
65     cdnss=0 ; cds=0
66     cembds=0 ; ces=0
67 End If
68
69 ! skip header record
70
71 Call get_line(safe,ntable,record)
72 If (.not.safe) Go To 100
73
74 ! read number of potential functions in file
75
76 Call get_line(safe,ntable,record)
77 If (.not.safe) Go To 100
78 Call get_word(record,word)

```

```

79      numpot = Nint(word_2_real(word))
80
81      Do ipot=1,numpot
82
83 ! read data type, atom labels, number of points, start and end
84
85      Call get_line(safe,ntable,record)
86      If (.not.safe) Go To 100
87
88 ! identify data type
89
90      Call get_word(record,keyword)
91      Call lower_case(keyword)
92      If (keyword == 'pair') Then
93          ktype = 1
94      Else If (keyword == 'dens' .or. keyword == 'dden') Then
95          ktype = 2
96      Else If (keyword == 'embe' .or. keyword == 'demb') Then
97          ktype = 3
98      Else If (keyword == 'sden') Then
99          ktype = 4
100     Else If (keyword == 'semb') Then
101         ktype = 5
102     Else
103         Call error(151)
104     End If
105
106 ! identify atom types
107
108     Call get_word(record,atom1)
109     If (ktype == 1 .or.                                     & ! pair
110         (ktype == 2 .and. (tabmet == 2 .or. tabmet == 4)) .or. & ! den for EEAM and dden for 2BEEAM
111         (ktype == 4 .and. (tabmet == 3 .or. tabmet == 4))) Then ! sden for 2B(EAM and EEAM)
112         Call get_word(record,atom2)
113     Else
114         atom2 = atom1
115     End If
116
117 ! data specifiers
118
119     Call get_word(record,word)
120     ngrid = Nint(word_2_real(word))
121     Call get_word(record,word)
122     start = word_2_real(word)
123     Call get_word(record,word)
124     finish = word_2_real(word)
125
126 ! check atom identities
127
128     katom1=0
129     katom2=0
130
131     Do jtpatm=1,ntpattm
132         If (atom1 == unqatm(jtpatm)) katom1=jtpatm
133         If (atom2 == unqatm(jtpatm)) katom2=jtpatm
134     End Do
135
136     If (katom1 == 0 .or. katom2 == 0) Then
137         If (idnode == 0 .and. l_top) &
138             Write(nrite,'(a)') '****',atom1,'***',atom2,'*** entry in TABEAM'
139         Call error(81)

```

```

140      End If
141
142 ! store working parameters
143
144 buffer(1)=Real(ngrid+4,wp) ! as if there are 4 extra elements after finish
145 buffer(4)=(finish-start)/Real(ngrid-1,wp)
146 buffer(2)=start-5.0_wp*buffer(4)
147 buffer(3)=finish
148
149 If (idnode == 0 .and. l_top) &
150   Write(nrite,"(1x,i10,4x,2a8,3x,2a4,2x,i6,1p,3e15.6)") &
151   ipot,atom1,atom2,'EAM-' ,keyword,ngrid,start,finish,buffer(4)
152
153 ! check array dimensions
154
155 If (ngrid+4 > mxgmet) Then
156   Call warning(270,Real(ngrid+4,wp),Real(mxgmet,wp),0.0_wp)
157   Call error(48)
158 End If
159
160 keymet=(Max(katom1,katom2)*(Max(katom1,katom2)-1))/2 + Min(katom1,katom2)
161 k0=lstmet(keymet)
162
163 ! check for undefined potential
164
165 If (k0 == 0) Call error(508)
166
167 ! read in potential arrays
168
169 Do i=1,(ngrid+3)/4
170   j=Min(4,ngrid-(i-1)*4)
171   If (idnode == 0) Then
172     Read(Unit=ntable, Fmt=*, End=100) buffer(4*i+1:4*i+j)
173   Else
174     buffer(4*i+1:4*i+j)=0.0_wp
175   End If
176 End Do
177
178 If (mxnode > 1) Call gsum(buffer(5:ngrid+4))
179
180 ! copy data to internal arrays
181
182 If (ktype == 1) Then
183
184 ! pair potential terms
185
186 ! Set indices
187
188 ! k0=lstmet(keymet)
189
190 cp=cp+1
191 If (Any(cpair(1:cp-1) == k0)) Then
192   Call error(509)
193 Else
194   cpair(cp)=k0
195 End If
196
197 vmet(1,k0,1)=buffer(1)
198 vmet(2,k0,1)=buffer(2)
199 vmet(3,k0,1)=buffer(3)
200 vmet(4,k0,1)=buffer(4)

```

```

201
202     Do i=5,mxgmet
203         If (i-4 > ngrid) Then
204             vmet(i,k0,1)=0.0_wp
205         Else
206             buffer(i)=buffer(i)*engunit
207             vmet(i,k0,1)=buffer(i)
208         End If
209     End Do
210
211 ! calculate derivative of pair potential function
212
213     Call metal_table_derivatives(k0,buffer,Size(vmet,2),vmet)
214
215 ! adapt derivatives for use in interpolation
216
217     Do i=5,ngrid+4
218         vmet(i,k0,2)=-(Real(i,wp)*buffer(4)+buffer(2))*vmet(i,k0,2)
219     End Do
220
221     Else If (ktype == 2) Then
222
223     ! density function terms
224     ! s-density density function terms for EAM & EEAM
225     ! d-density density function terms for 2B(EAM & EEAM)
226
227     ! Set indices
228
229     If      (tabmet == 1 .or. tabmet == 3) Then ! EAM
230         k0=katom1
231     Else If (tabmet == 2 .or. tabmet == 4) Then ! EEAM
232         k0=(katom1-1)*ntpatm+katom2
233     End If
234
235     cd=cd+1
236     If (Any(cdens(1:cd-1) == k0)) Then
237         Call error(510)
238     Else
239         cdens(cd)=k0
240     End If
241
242     dmet(1,k0,1)=buffer(1)
243     dmet(2,k0,1)=buffer(2)
244     dmet(3,k0,1)=buffer(3)
245     dmet(4,k0,1)=buffer(4)
246
247     Do i=5,mxgmet
248         If (i-4 > ngrid) Then
249             dmet(i,k0,1)=0.0_wp
250         Else
251             dmet(i,k0,1)=buffer(i)
252         End If
253     End Do
254
255     ! calculate derivative of density function
256
257     Call metal_table_derivatives(k0,buffer,Size(dmet,2),dmet)
258
259     ! adapt derivatives for use in interpolation
260
261     dmet(1,k0,2)=0.0_wp

```

```

262     dmet(2,k0,2)=0.0_wp
263     dmet(3,k0,2)=0.0_wp
264     dmet(4,k0,2)=0.0_wp
265
266     Do i=5,ngrid+4
267       dmet(i,k0,2)=-(Real(i,wp)*buffer(4)+buffer(2))*dmet(i,k0,2)
268     End Do
269
270   Else If (ktype == 3) Then
271
272   ! embedding function terms
273   ! s-density embedding function terms for EAM & EEAM
274   ! d-density embedding function terms for 2B(EAM & EEAM)
275
276   ! Set indices
277
278   k0=katom1
279
280   ce=ce+1
281   If (Any(cembed(1:ce-1) == k0)) Then
282     Call error(511)
283   Else
284     cembed(ce)=k0
285   End If
286
287   fmet(1,k0,1)=buffer(1)
288   fmet(2,k0,1)=buffer(2)
289   fmet(3,k0,1)=buffer(3)
290   fmet(4,k0,1)=buffer(4)
291
292   Do i=5,mxgmet
293     If (i-4 > ngrid) Then
294       fmet(i,k0,1)=0.0_wp
295     Else
296       buffer(i)=buffer(i)*engunit
297       fmet(i,k0,1)=buffer(i)
298     End If
299   End Do
300
301   ! calculate derivative of embedding function
302
303   Call metal_table_derivatives(k0,buffer,Size(fmet,2),fmet)
304
305   Else If (ktype == 4) Then
306
307   ! s-density function terms
308
309   ! The 2BM formalism for alloys allows for a mixed s-band density: rho_{atom1,atom2} /= 0
310   ! (and in general for the EEAM it may be non-symmetric: rho_{atom1,atom2} may be /= rho_{atom2,atom1})
311   ! Some 2BM models rho_{atom1,atom1}=rho_{atom2,atom2}=0 with rho_{atom1,atom2} /= 0
312   ! whereas others choose not to have mixed s-band densities.
313
314   ! Set indices
315
316   If (tabmet == 3) Then ! 2BMEAM
317     k0=lstmet(keymet)
318   Else If (tabmet == 4) Then ! 2BMEEAM
319     k0=(katom1-1)*ntpam+katom2
320   End If
321
322   cds=cds+1

```

```

323     If (Any(cdnss(1:cds-1) == k0)) Then
324         Call error(510)
325     Else
326         cdnss(cds)=k0
327     End If
328
329     dmes(1,k0,1)=buffer(1)
330     dmes(2,k0,1)=buffer(2)
331     dmes(3,k0,1)=buffer(3)
332     dmes(4,k0,1)=buffer(4)
333
334     If (Nint(buffer(1)) > 5) Then
335
336         Do i=5,mxgmet
337             If (i-4 > ngrid) Then
338                 dmes(i,k0,1)=0.0_wp
339             Else
340                 dmes(i,k0,1)=buffer(i)
341             End If
342         End Do
343 ! calculate derivative of density function
344
345     Call metal_table_derivatives(k0,buffer,Size(dmes,2),dmes)
346
347 ! adapt derivatives for use in interpolation
348
349     dmes(1,k0,2)=0.0_wp
350     dmes(2,k0,2)=0.0_wp
351     dmes(3,k0,2)=0.0_wp
352     dmes(4,k0,2)=0.0_wp
353
354     Do i=5,ngrid+4
355         dmes(i,k0,2)=-(Real(i,wp)*buffer(4)+buffer(2))*dmes(i,k0,2)
356     End Do
357
358     End If
359
360     Else If (ktype == 5) Then
361
362 ! s-embedding function terms
363
364 ! Set index
365
366     k0=katom1
367
368     ces=ces+1
369     If (Any(cembds(1:ces-1) == k0)) Then
370         Call error(511)
371     Else
372         cembds(ces)=k0
373     End If
374
375     fmcs(1,k0,1)=buffer(1)
376     fmcs(2,k0,1)=buffer(2)
377     fmcs(3,k0,1)=buffer(3)
378     fmcs(4,k0,1)=buffer(4)
379
380     Do i=5,mxgmet
381         If (i-4 > ngrid) Then
382             fmcs(i,k0,1)=0.0_wp
383         Else

```

```

384         buffer(i)=buffer(i)*engunit
385         fmes(i,k0,1)=buffer(i)
386         End If
387     End Do
388
389 ! calculate derivative of embedding function
390
391     Call metal_table_derivatives(k0,buffer,Size(fmes,2),fmes)
392
393     End If
394
395 End Do
396
397 If (idnode == 0) Close(Unit=ntable)
398 If (idnode == 0 .and. l_top) Write(nrite,'(/,1x,a)') 'potential tables read from TABEAM file'
399
400 If      (tabmet == 1 .or. tabmet == 2) Then ! EAM & EEAM
401     Deallocate (cpair,cdens,cembed,           Stat=fail(1))
402 Else If (tabmet == 3 .or. tabmet == 4) Then ! 2B(EAM & EEAM)
403     Deallocate (cpair,cdens,cdnss,cembed,cembds, Stat=fail(1))
404 End If
405 Deallocate (buffer,                         Stat=fail(2))
406 If (Any(fail > 0)) Then
407     Write(nrite,'(/,1x,a,i0)') 'metal_table_read deallocation failure, node: ', idnode
408     Call error(0)
409 End If
410
411 Return
412
413 ! end of file error exit
414
415 100 Continue
416
417 If (idnode == 0) Close(Unit=ntable)
418 Call error(24)
419
420 End Subroutine metal_table_read

```
