Preliminaries

```
Clear["Global`*"]

AbundSeries = "GoldenRatio";

Which[
  AbundSeries == "GoldenRatio",
   SetDirectory[FileNameJoin[{NotebookDirectory[], "../results/GoldenRatio/"}]] ,
  AbundSeries == "Arithmetic",
   SetDirectory[FileNameJoin[{NotebookDirectory[], "../results/Arithmetic/"}]]
 ];
```

Define functions to compute fisher matrix and expected fisher matrix

Definition using Hessian (applies under mild regularity conditions):

In[•]:=
```
fisher[fun_, pars_] := -D[Apply[fun, pars], {pars, 2}]
```

Expected Fisher Matrix given experimental design (*unused*)

In[•]:=
```
EFisher[FisherMatrix_, func_, PDistE_, NDistE_, subs_] :=
 Expectation[
  Expectation[
   FisherMatrix /. subs, x ≈ PoissonDistribution[func /. subs]],
  {P ≈ PDistE, N ≈ NDistE}]
```

Min and max limits on expected count of prey eaten

```
In[•]:= SqrtDetEFlim[func_, Det_, Nvals_, Pvals_, subs_, FminMult_, FmaxMult_] :=
          Piecewise[
           {{Sqrt[Det],
             Max[func /. subs /. P → Pvals /. N → Nvals] ≤ Max[Nvals] * FmaxMult &&
              Min[func /. subs /. P → Pvals /. N → {Max[Nvals]}] ≥ 1 * FminMult
            }}];
        SqrtDetEFlimSN1[func_, Det_, Nvals_, Pvals_, subs_, FminMult_, FmaxMult_] :=
          Piecewise[
           {{Sqrt[Det],
             Max[func /. subs /. P → Pvals /. N → Nvals] ≤ Max[Nvals] * FmaxMult &&
              Min[func /. subs /. P → Pvals /. N → {Max[Nvals]}] ≥ 1 * FminMult &&
              d b Max[func /. subs /. P → Pvals /. N → Nvals] ≤ 1
            }}];
        SqrtDetEFlimSN2[func_, Det_, Nvals_, Pvals_, subs_, FminMult_, FmaxMult_] :=
          Piecewise[
           {{Sqrt[Det],
             Max[func /. subs /. P → Pvals /. N → Nvals] ≤ Max[Nvals] * FmaxMult &&
              Min[func /. subs /. P → Pvals /. N → {Max[Nvals]}] ≥ 1 * FminMult &&
              b Max[func /. subs /. P → Pvals /. N → Nvals] ≤ 1
            }}];
```

Define Functional Responses - Count of prey eaten

```
In[•]:= (* k=1 *)
        H1 = a N P T; (* Holling Type I *)
        LR = a N/P P T; (* Linear ratio-dependent *)
        BWL1 = a √N/√P P T; (* Barbier,Wojcik & Loreau 2021 *)

        (*k=2*)
        H2 = a N/(1 + a b N) P T; (* Holling Type II *)
        MM = a N/(b + N) P T; (* Michaelis-Menten *)
        HV = a N/P^v P T; (* Hassell-Varley *)
        R = a N^u P T; (* Rosenzweig '71 *)
        AG = a N/P/(1 + a b N/P) P T; (* Arditi-Ginzburg, Sutherland *)
```

$$\text{CDAO} = \frac{a\,\frac{N}{\sqrt{P}}}{1 + a\,b\,\frac{N}{\sqrt{P}}}\,P\,T; \quad (* \text{ Cosner et al. 1999 } *)$$

$$\text{GI} = \frac{1}{b}\,(1 - \text{Exp}[-a\,N])\,P\,T;$$

$$\text{GIA} = \frac{1}{b}\,(1 - \text{Exp}[-a\,b\,N])\,P\,T; \quad (* \text{ Gause-Ivlev modified by Aldebert } *)$$

$$\text{GB} = \frac{1}{b}\,(1 - \text{Exp}[-a\,N / P])\,P\,T; \quad (* \text{ Gutierrez \& Baumgaertner 1984 } *)$$

$$\text{HT} = \frac{1}{b}\,\text{Tanh}[a\,b\,N]\,P\,T; \quad (* \text{ Jassby \& Platt 1976 } *)$$

$$\text{HTb} = \frac{1}{b}\,\frac{\text{Exp}[2\,a\,b\,N] - 1}{\text{Exp}[2\,a\,b\,N] + 1}\,P\,T;$$

(* Alternative parameterization of Jassby & Platt 1976 *)

$$\text{H3} = \frac{a\,N^2}{1 + a\,b\,N^2}\,P\,T; \quad (* \text{ Holling Type III } *)$$

$$\text{AGK} = \frac{a\,\left(\frac{N}{P}\right)^2}{1 + a\,b\,\left(\frac{N}{P}\right)^2}\,P\,T; \quad (* \text{ Kratina et al. 2008 } *)$$

$$\text{A0} = \frac{a\,N}{1 + a\,b\,\sqrt{N}}\,P\,T; (* \ \frac{a\,N}{1 + \sqrt{a\,b\,N}}\,P\ T; \ *) \quad (* \text{ Abrams 1982 } *)$$

$$\text{A1} = \sqrt{\frac{a\,N}{1 + a\,b\,N}}\ P\,T; (* \ \sqrt{\frac{q\,c\,N}{d(1 + c\,h\,N)}}\,P\ T\ *) \quad (* \text{ Abrams 1990 } *)$$

$$\text{A3} = \frac{a\,\sqrt{N}}{1 + a\,b\,\sqrt{N}}\,P\,T; (* \frac{\sqrt{N}}{2\sqrt{u\,\eta} + h\,\sqrt{N}}\ P\ T\ *) (* \text{ Abrams 1990 } *)$$

$$\text{SH} = \frac{a\,N}{b + N^2}\,P\,T; \quad (* \text{ Sokol \& Howell 1987 } *)$$

$$\text{SHb} = \frac{a\,N}{1 + a\,b\,N^2}\,P\,T; \quad (* \text{ Holling-form Sokol \& Howell 1987 } *)$$

(*k=3*)

$$\text{BD} = \frac{a\,N}{1 + a\,b\,N + c\,(P - 1)}\,P\,T; \quad (* \text{ Beddington-DeAngelis } *)$$

$$\text{CM} = \frac{a\,N}{1 + a\,b\,N + c\,(P - 1) + a\,b\,c\,N\,(P - 1)}\,P\,T; \quad (* \text{ Crowley-Martin } *)$$

$$\text{AA} = \frac{a\,\frac{N}{P^v}}{1 + a\,b\,\frac{N}{P^v}}\,P\,T; \quad (* \text{ Arditi-Akcakaya } *)$$

$$\text{BWL2} = a\,N^u\,P^v\,T; \quad (* \text{ Barbier,Wojcik \& Loreau 2021 } *)$$

$$H3R = \frac{a\,N^{u}}{1 + a\,b\,N^{u}}\,P\,T;\ (*\ \text{Holling Type III}\ *)$$

$$A2 = \frac{a\,N}{1 + a\,b\,N + \sqrt{a\,N\,c\,(1 + a\,b\,N)}}\,P\,T;\ (*\ \text{Abrams 1990}\ *)$$

$$S3 = \left(\frac{a\,N}{1 + a\,b\,N}\right)^{u}\,P\,T;\ (*\ \text{new, generalized A1}\ *)$$

$$S3b = \left(\frac{a\,N}{b + N}\right)^{u}\,P\,T;\ (*\ \text{new, generalized A1 in MM form}*)$$

$$HLB = \frac{a\,N^{2}}{1 + c\,N + a\,b\,N^{2}}\,P\,T;\ (*\ \text{Hassell, Lawton \& Beddington 1977}\ *)$$

$$HLBb = \frac{a\,N^{2}}{b + N + c\,N^{2}}\,P\,T;\ (*\ \text{Michaelis-Mentend-form of Hassell,}$$
$$\text{Lawton \& Beddington 1977}\ *)$$

$$MH = \frac{a\,N}{1 + a\,b\,N + c\,N^{2}}\,P\,T;\ (*\ \text{Holling-form Monod-Haldane; Andrews 1968}\ *)$$

$$MHb = \frac{a\,N}{b + N + c\,N^{2}}\,P\,T;\ (*\ \text{Monod-Haldane; Andrews 1968}\ *)$$

$$MHc = \frac{a\,N}{b + N + \frac{N^{2}}{c}}\,P\,T;\ (*\ \text{original Monod-Haldane; Andrews 1968}\ *)$$

$$To = \frac{a\,N}{1 + a\,b\,N + c\,N^{3}}\,P\,T;\ (*\ \text{Tostowaryk '72}\ *)$$

$$FHM = \frac{a\,N\,\text{Exp}[d\,N]}{1 + a\,b\,N\,\text{Exp}[d\,N]}\,P\,T;\ (*\ \text{Fujii, Holling \& Mace '86}\ *)$$

$$W = \frac{1}{b}\left(1 - \text{Exp}\left[-a\,N \,/\, P^{v}\right]\right)P\,T;\ (*\ \text{Watt 1959}\ *)$$

$$TTA = \frac{a\,N}{1 + a\,b\,N + c\,P - (1 - \text{Exp}[-c\,P])}\,P\,T;\ (*\ \text{Tyutyunov, Titova \& Arditi 2008}\ *)$$

$$SBB = \frac{a\left(\frac{N}{P^{v}}\right)^{2}}{1 + a\,b\left(\frac{N}{P^{v}}\right)^{2}}\,P\,T;\ \ (*\ \text{Schenk, Bersier \& Bacher 2005}\ *)$$

$$SSS = \frac{2\,a\,N}{1 + a\,(b + c)\,N + \sqrt{(1 + a\,(b + c)\,N)\,(1 + a\,(b + c + 4\,b\,c)\,N)}}\,P\,T;$$
$$(*\ \text{Jeschke et al. 2002 using citardauq Formula}\ *)$$

$$RGD = \frac{2\,a\,N}{1 + a\,b\,N + \sqrt{(1 + a\,b\,N)^{2} + 8\,a\,c\,(P - 1)}}\,P\,T;$$

(*k=4*)

```
BDOR = (a N^u) / (1 + a b N^u + c (P - 1)) P T; (* Okuyama & Ruyle 2011 *)

CMOR = (a N^u) / (1 + a b N^u + c (P - 1) + a b c N^u (P - 1)) P T; (* Okuyama & Ruyle 2011 *)

AAOR = (a N^u / P^v) / (1 + a b N^u / P^v) P T; (* Okuyama & Ruyle 2011 *)

SN1 = (a N) / (1 + a b N + c (P - 1) + a b c (1 - d) N (P - 1)) P T; (* Stouffer & Novak 2021 *)

SN2 = (a N (1 + c (1 - d) (P - 1))) / (1 + a b N + c (P - 1) + a b c (1 - d) N (P - 1)) P T;
(* new, but see Stouffer & Novak 2021 *)
```

```
models = {
   H1, LR, BWL1,
   H2, MM, HV, R, AG, CDAO, GI, GIA, GB, HT, HTb, H3, AGK, A0, A1, A3, SH, SHb,
   BD, CM, AA, BWL2, H3R, A2, S3, S3b,
   HLB, HLBb, MH, MHb, MHc, To, FHM, W, TTA, SBB, SSS, RGD,
   BDOR, CMOR, AAOR, SN1, SN2};
modelNames = (Trace@{
        H1, LR, BWL1,
        H2, MM, HV, R, AG, CDAO, GI,
        GIA, GB, HT, HTb, H3, AGK, A0, A1, A3, SH, SHb,
        BD, CM, AA, BWL2, H3R, A2, S3, S3b, HLB, HLBb, MH,
        MHb, MHc, To, FHM, W, TTA, SBB, SSS, RGD,
        BDOR, CMOR, AAOR, SN1, SN2})[[All, 1]][[1 ;; Length[models]]];
DumpSave["../Models.mx",
  {models, modelNames,
   H1, LR, BWL1,
   H2, MM, HV, R, AG, CDAO, GI, GIA, GB, HT, HTb, H3, AGK, A0, A1, A3, SH, SHb,
   BD, CM, AA, BWL2, H3R, A2, S3, S3b,
   HLB, HLBb, MH, MHb, MHc, To, FHM, W, TTA, SBB, SSS, RGD,
   BDOR, CMOR, AAOR, SN1, SN2}];
```

Method to assess whether model parameters are identifiable (a “necessary but not sufficient” test)

```
(*
model=(a N)/(c+a b N) P T;
parms={a, b,c};
sensitivity=D[model,{parms}]
GatherBy[Range@Length[sensitivity],sensitivity[[#]]&]
*)
```

Define Likelihood functions

```
PoislL[func_] := -n func + Log[func] x /. {n → 1}

lLH1[a_] := PoislL[H1]
lLLR[a_] := PoislL[LR]
lLBWL1[a_] := PoislL[BWL1]

lLH2[a_, b_] := PoislL[H2]
lLMM[a_, b_] := PoislL[MM]
lLHV[a_, v_] := PoislL[HV]
lLR[a_, u_] := PoislL[R]
lLAG[a_, b_] := PoislL[AG]
lLCDAO[a_, b_] := PoislL[CDAO]
lLGI[a_, b_] := PoislL[GI]
lLGIA[a_, b_] := PoislL[GIA]
lLGB[a_, b_] := PoislL[GB]
lLHT[a_, b_] := PoislL[HT]
lLHTb[a_, b_] := PoislL[HTb]
lLH3[a_, b_] := PoislL[H3]
lLAGK[a_, b_] := PoislL[AGK]
lLA0[a_, b_] := PoislL[A0]
lLA3[a_, b_] := PoislL[A3]
lLA1[a_, b_] := PoislL[A1]
lLSH[a_, b_] := PoislL[SH]
lLSHb[a_, b_] := PoislL[SHb]

lLBD[a_, b_, c_] := PoislL[BD]
lLCM[a_, b_, c_] := PoislL[CM]
lLAA[a_, b_, v_] := PoislL[AA]
lLBWL2[a_, v_, u_] := PoislL[BWL2]
lLH3R[a_, b_, u_] := PoislL[H3R]
lLA2[a_, b_, c_] := PoislL[A2]
lLHLB[a_, b_, c_] := PoislL[HLB]
lLHLBb[a_, b_, c_] := PoislL[HLBb]
```

```
lLMH[a_, b_, c_] := PoislL[MH]
lLMHb[a_, b_, c_] := PoislL[MHb]
lLMHc[a_, b_, c_] := PoislL[MHc]
lLTo[a_, b_, c_] := PoislL[To]
lLFHM[a_, b_, d_] := PoislL[FHM]
lLW[a_, b_, v_] := PoislL[W]
lLTTA[a_, b_, c_] := PoislL[TTA]
lLSBB[a_, b_, v_] := PoislL[SBB]
lLSSS[a_, b_, c_] := PoislL[SSS]
lLRGD[a_, b_, c_] := PoislL[RGD]
lLS3[a_, b_, u_] := PoislL[S3]
lLS3b[a_, b_, u_] := PoislL[S3b]

lLBDOR[a_, b_, c_, u_] := PoislL[BDOR]
lLCMOR[a_, b_, c_, u_] := PoislL[CMOR]
lLAAOR[a_, b_, v_, u_] := PoislL[AAOR]
lLSN1[a_, b_, c_, d_] := PoislL[SN1]
lLSN2[a_, b_, c_, d_] := PoislL[SN2]
```

Define master Geometric Complexity function

```
ClearAll[GeomComplex]
GeomComplex[
  Nvalues_,
  Pvalues_,
  Model_] :=
 Module[
  {
   Nvals = Nvalues,
   Pvals = Pvalues,
   Tval = 1,
   NIntMethod = {"LocalAdaptive", "SingularityHandler" → Automatic},
   minRec = 150,
   maxRec = 500,
   accgoal = 3,
   precgoal = 3,
   FminMult = 1, (* 0.1 or 1 *)
   FmaxMult = 1 (* 10 or 1 *)
  },
  Nprobs = ConstantArray[1 / Length[Nvals], Length[Nvals]];
  Pprobs = ConstantArray[1 / Length[Pvals], Length[Pvals]];
  NDistE = EmpiricalDistribution[Nprobs → Nvals];
  PDistE = EmpiricalDistribution[Pprobs → Pvals];
```

```
subs = {T → Tval};

ParmRange = {
  {a, 0, Infinity},
  {b, 0, Infinity},
  {c, 0, Infinity},
  {v, 0, 1, Infinity},
  {u, 0, 1, Infinity},
  {d, -Infinity, -1, 0, 1, Infinity}
 };

Which[
 Model == "H1",
 DetH1 = Det[EFisher[fisher[lLH1, {a}], H1, PDistE, NDistE, subs]];
 NIntH1 =
  Log[
   NIntegrate[
    SqrtDetEFlim[H1, DetH1, Nvals, Pvals, subs, FminMult, FmaxMult],
    ParmRange[[1]],
    AccuracyGoal → accgoal,
    PrecisionGoal → precgoal,
    Method → NIntMethod,
    MaxRecursion → maxRec]]
 ,
 Model == "LR",
 DetLR = Det[EFisher[fisher[lLLR, {a}], LR, PDistE, NDistE, subs]];
 NIntLR =
  Log[
   NIntegrate[
    SqrtDetEFlim[LR, DetLR, Nvals, Pvals, subs, FminMult, FmaxMult],
    ParmRange[[1]],
    AccuracyGoal → accgoal,
    PrecisionGoal → precgoal,
    Method → NIntMethod,
    MaxRecursion → maxRec]]
 ,
 Model == "BWL1",
 DetBWL1 = Det[EFisher[fisher[lLBWL1, {a}], BWL1, PDistE, NDistE, subs]];
 NIntBWL1 =
  Log[
   NIntegrate[
    SqrtDetEFlim[BWL1, DetBWL1, Nvals, Pvals, subs, FminMult, FmaxMult],
    ParmRange[[1]],
    AccuracyGoal → accgoal,
```

```
    PrecisionGoal → precgoal,
    Method → NIntMethod,
    MaxRecursion → maxRec]]
,
Model == "H2",
DetH2 = Det[EFisher[fisher[lLH2, {a, b}], H2, PDistE, NDistE, subs]];
NIntH2 =
 Log[
  NIntegrate[
   SqrtDetEFlim[H2, DetH2, Nvals, Pvals, subs, FminMult, FmaxMult],
   ParmRange[[1]],
   ParmRange[[2]],
   AccuracyGoal → accgoal,
   PrecisionGoal → precgoal,
   Method → NIntMethod,
   MinRecursion → minRec,
   MaxRecursion → maxRec]]
,
Model == "MM",
DetMM = Det[EFisher[fisher[lLMM, {a, b}], MM, PDistE, NDistE, subs]];
NIntMM =
 Log[
  NIntegrate[
   SqrtDetEFlim[MM, DetMM, Nvals, Pvals, subs, FminMult, FmaxMult],
   ParmRange[[1]],
   ParmRange[[2]],
   AccuracyGoal → accgoal,
   PrecisionGoal → precgoal,
   Method → NIntMethod,
   MinRecursion → minRec,
   MaxRecursion → maxRec]]
,
Model == "HV",
DetHV = Det[EFisher[fisher[lLHV, {a, v}], HV, PDistE, NDistE, subs]];
NIntHV =
 Log[
  NIntegrate[
   SqrtDetEFlim[HV, DetHV, Nvals, Pvals, subs, FminMult, FmaxMult],
   ParmRange[[1]],
   ParmRange[[4]],
   AccuracyGoal → accgoal,
   PrecisionGoal → precgoal,
   Method → NIntMethod,
   MinRecursion → minRec,
```

```
    MaxRecursion → maxRec]]
,
Model == "R",
DetR = Det[EFisher[fisher[lLR, {a, u}], R, PDistE, NDistE, subs]];
NIntR =
 Log[
  NIntegrate[
   SqrtDetEFlim[R, DetR, Nvals, Pvals, subs, FminMult, FmaxMult],
   ParmRange[[1]],
   ParmRange[[5]],
   AccuracyGoal → accgoal,
   PrecisionGoal → precgoal,
   Method → NIntMethod,
   MinRecursion → minRec,
   MaxRecursion → maxRec]]
,
Model == "H3",
DetH3 = Det[EFisher[fisher[lLH3, {a, b}], H3, PDistE, NDistE, subs]];
NIntH3 =
 Log[
  NIntegrate[
   SqrtDetEFlim[H3, DetH3, Nvals, Pvals, subs, FminMult, FmaxMult],
   ParmRange[[1]],
   ParmRange[[2]],
   AccuracyGoal → accgoal,
   PrecisionGoal → precgoal,
   Method → NIntMethod,
   MinRecursion → minRec,
   MaxRecursion → maxRec]]
,
Model == "AG",
DetAG = Det[EFisher[fisher[lLAG, {a, b}], AG, PDistE, NDistE, subs]];
NIntAG =
 Log[
  NIntegrate[
   SqrtDetEFlim[AG, DetAG, Nvals, Pvals, subs, FminMult, FmaxMult],
   ParmRange[[1]],
   ParmRange[[2]],
   AccuracyGoal → accgoal,
   PrecisionGoal → precgoal,
   Method → NIntMethod,
   MinRecursion → minRec,
   MaxRecursion → maxRec]]
,
```

```
Model == "CDAO",
DetCDAO = Det[EFisher[fisher[lLCDAO, {a, b}], CDAO, PDistE, NDistE, subs]];
NIntCDAO =
 Log[
  NIntegrate[
   SqrtDetEFlim[CDAO, DetCDAO, Nvals, Pvals, subs, FminMult, FmaxMult],
   ParmRange[[1]],
   ParmRange[[2]],
   AccuracyGoal → accgoal,
   PrecisionGoal → precgoal,
   Method → NIntMethod,
   MinRecursion → minRec,
   MaxRecursion → maxRec]]
,
Model == "GI",
DetGI = Det[EFisher[fisher[lLGI, {a, b}], GI, PDistE, NDistE, subs]];
NIntGI =
 Log[
  NIntegrate[
   SqrtDetEFlim[GI, DetGI, Nvals, Pvals, subs, FminMult, FmaxMult],
   ParmRange[[1]],
   ParmRange[[2]],
   AccuracyGoal → accgoal,
   PrecisionGoal → precgoal,
   Method → NIntMethod,
   MinRecursion → 0,
   MaxRecursion → maxRec]]
,
Model == "GIA",
DetGIA = Det[EFisher[fisher[lLGIA, {a, b}], GIA, PDistE, NDistE, subs]];
NIntGIA =
 Log[
  NIntegrate[
   SqrtDetEFlim[GIA, DetGIA, Nvals, Pvals, subs, FminMult, FmaxMult],
   ParmRange[[1]],
   ParmRange[[2]],
   AccuracyGoal → accgoal,
   PrecisionGoal → precgoal,
   Method → NIntMethod,
   MinRecursion → minRec,
   MaxRecursion → maxRec]]
,
Model == "GB",
DetGB = Det[EFisher[fisher[lLGB, {a, b}], GB, PDistE, NDistE, subs]];
```

```
NIntGB =
 Log[
  NIntegrate[
   SqrtDetEFlim[GB, DetGB, Nvals, Pvals, subs, FminMult, FmaxMult],
   ParmRange[[1]],
   ParmRange[[2]],
   AccuracyGoal → accgoal,
   PrecisionGoal → precgoal,
   Method → NIntMethod,
   MinRecursion → 0,
   MaxRecursion → maxRec]]
,
Model == "HT",
DetHT = Det[EFisher[fisher[lLHT, {a, b}], HT, PDistE, NDistE, subs]];
NIntHT =
 Log[
  NIntegrate[
   SqrtDetEFlim[HT, DetHT, Nvals, Pvals, subs, FminMult, FmaxMult],
   ParmRange[[1]],
   ParmRange[[2]],
   AccuracyGoal → accgoal,
   PrecisionGoal → precgoal,
   Method → NIntMethod,
   MinRecursion → minRec,
   MaxRecursion → maxRec]]
,
Model == "HTb",
DetHTb = Det[EFisher[fisher[lLHTb, {a, b}], HTb, PDistE, NDistE, subs]];
NIntHTb =
 Log[
  NIntegrate[
   SqrtDetEFlim[HTb, DetHTb, Nvals, Pvals, subs, FminMult, FmaxMult],
   ParmRange[[1]],
   ParmRange[[2]],
   AccuracyGoal → accgoal,
   PrecisionGoal → precgoal,
   Method → NIntMethod,
   MinRecursion → minRec,
   MaxRecursion → maxRec]]
,
Model == "A0",
DetA0 = Det[EFisher[fisher[lLA0, {a, b}], A0, PDistE, NDistE, subs]];
NIntA0 =
 Log[
```

```
  NIntegrate[
   SqrtDetEFlim[A0, DetA0, Nvals, Pvals, subs, FminMult, FmaxMult],
   ParmRange[[1]],
   ParmRange[[2]],
   AccuracyGoal → accgoal,
   PrecisionGoal → precgoal,
   Method → NIntMethod,
   MinRecursion → minRec,
   MaxRecursion → maxRec]]
,
Model == "A3",
DetA3 = Det[EFisher[fisher[lLA3, {a, b}], A3, PDistE, NDistE, subs]];
NIntA3 =
 Log[
  NIntegrate[
   SqrtDetEFlim[A3, DetA3, Nvals, Pvals, subs, FminMult, FmaxMult],
   ParmRange[[1]],
   ParmRange[[2]],
   AccuracyGoal → accgoal,
   PrecisionGoal → precgoal,
   Method → NIntMethod,
   MinRecursion → minRec,
   MaxRecursion → maxRec]]
,
Model == "AGK",
DetAGK = Det[EFisher[fisher[lLAGK, {a, b}], AGK, PDistE, NDistE, subs]];
NIntAGK =
 Log[
  NIntegrate[
   SqrtDetEFlim[AGK, DetAGK, Nvals, Pvals, subs, FminMult, FmaxMult],
   ParmRange[[1]],
   ParmRange[[2]],
   AccuracyGoal → accgoal,
   PrecisionGoal → precgoal,
   Method → NIntMethod,
   MinRecursion → minRec,
   MaxRecursion → maxRec]]
,
Model == "A1",
DetA1 = Det[EFisher[fisher[lLA1, {a, b}], A1, PDistE, NDistE, subs]];
NIntA1 =
 Log[
  NIntegrate[
   SqrtDetEFlim[A1, DetA1, Nvals, Pvals, subs, FminMult, FmaxMult],
```

```
    ParmRange[[1]],
    ParmRange[[2]],
    AccuracyGoal → accgoal,
    PrecisionGoal → precgoal,
    Method → NIntMethod,
    MinRecursion → minRec,
    MaxRecursion → maxRec]]
,
Model == "SH",
DetSH = Det[EFisher[fisher[lLSH, {a, b}], SH, PDistE, NDistE, subs]];
NIntSH =
 Log[
  NIntegrate[
   SqrtDetEFlim[SH, DetSH, Nvals, Pvals, subs, FminMult, FmaxMult],
   ParmRange[[1]],
   ParmRange[[2]],
   AccuracyGoal → accgoal,
   PrecisionGoal → precgoal,
   Method → NIntMethod,
   MinRecursion → minRec,
   MaxRecursion → maxRec] ]
,
Model == "SHb",
DetSHb = Det[EFisher[fisher[lLSHb, {a, b}], SHb, PDistE, NDistE, subs]];
NIntSHb =
 Log[
  NIntegrate[
   SqrtDetEFlim[SHb, DetSHb, Nvals, Pvals, subs, FminMult, FmaxMult],
   ParmRange[[1]],
   ParmRange[[2]],
   AccuracyGoal → accgoal,
   PrecisionGoal → precgoal,
   Method → NIntMethod,
   MinRecursion → minRec,
   MaxRecursion → maxRec] ]
,
Model == "BD",
DetBD = Det[EFisher[fisher[lLBD, {a, b, c}], BD, PDistE, NDistE, subs]];
NIntBD =
 Log[
  NIntegrate[
   SqrtDetEFlim[BD, DetBD, Nvals, Pvals, subs, FminMult, FmaxMult],
   ParmRange[[1]],
   ParmRange[[2]],
```

```
    ParmRange[[3]],
    AccuracyGoal → accgoal,
    PrecisionGoal → precgoal,
    Method → NIntMethod,
    MinRecursion → minRec,
    MaxRecursion → maxRec]]
,
Model == "CM",
DetCM = Det[EFisher[fisher[lLCM, {a, b, c}], CM, PDistE, NDistE, subs]];
NIntCM =
 Log[
  NIntegrate[
   SqrtDetEFlim[CM, DetCM, Nvals, Pvals, subs, FminMult, FmaxMult],
   ParmRange[[1]],
   ParmRange[[2]],
   ParmRange[[3]],
   AccuracyGoal → accgoal,
   PrecisionGoal → precgoal,
   Method → NIntMethod,
   MinRecursion → minRec,
   MaxRecursion → maxRec]]
,
Model == "AA",
DetAA = Det[EFisher[fisher[lLAA, {a, b, v}], AA, PDistE, NDistE, subs]];
NIntAA =
 Log[
  NIntegrate[
   SqrtDetEFlim[AA, DetAA, Nvals, Pvals, subs, FminMult, FmaxMult],
   ParmRange[[1]],
   ParmRange[[2]],
   ParmRange[[4]],
   AccuracyGoal → accgoal,
   PrecisionGoal → precgoal,
   Method → NIntMethod,
   MinRecursion → minRec,
   MaxRecursion → maxRec]]
,
Model == "BWL2",
DetBWL2 = Det[EFisher[fisher[lLBWL2, {a, v, u}], BWL2, PDistE, NDistE, subs]];
NIntBWL2 =
 Log[
  NIntegrate[
   SqrtDetEFlim[BWL2, DetBWL2, Nvals, Pvals, subs, FminMult, FmaxMult],
   ParmRange[[1]],
```

```
    ParmRange[[4]],
    ParmRange[[5]],
    AccuracyGoal → accgoal,
    PrecisionGoal → precgoal,
    Method → NIntMethod,
    MinRecursion → minRec,
    MaxRecursion → maxRec]]
,
Model == "H3R",
DetH3R = Det[EFisher[fisher[lLH3R, {a, b, u}], H3R, PDistE, NDistE, subs]];
NIntH3R =
 Log[
  NIntegrate[
   SqrtDetEFlim[H3R, DetH3R, Nvals, Pvals, subs, FminMult, FmaxMult],
   ParmRange[[1]],
   ParmRange[[2]],
   ParmRange[[5]],
   AccuracyGoal → accgoal,
   PrecisionGoal → precgoal,
   Method → NIntMethod,
   MinRecursion → minRec,
   MaxRecursion → maxRec]]
,
Model == "A2",
DetA2 = Det[EFisher[fisher[lLA2, {a, b, c}], A2, PDistE, NDistE, subs]];
NIntA2 =
 Log[
  NIntegrate[
   SqrtDetEFlim[A2, DetA2, Nvals, Pvals, subs, FminMult, FmaxMult],
   ParmRange[[1]],
   ParmRange[[2]],
   ParmRange[[3]],
   AccuracyGoal → accgoal,
   PrecisionGoal → precgoal,
   Method → NIntMethod,
   MinRecursion → minRec,
   MaxRecursion → maxRec]]
,
Model == "HLB",
DetHLB = Det[EFisher[fisher[lLHLB, {a, b, c}], HLB, PDistE, NDistE, subs]];
NIntHLB =
 Log[
  NIntegrate[
   SqrtDetEFlim[HLB, DetHLB, Nvals, Pvals, subs, FminMult, FmaxMult],
```

```
    ParmRange[[1]],
    ParmRange[[2]],
    ParmRange[[3]],
    AccuracyGoal → accgoal,
    PrecisionGoal → precgoal,
    Method → NIntMethod,
    MinRecursion → minRec,
    MaxRecursion → maxRec]]
,
Model == "HLBb",
DetHLBb = Det[EFisher[fisher[lLHLBb, {a, b, c}], HLBb, PDistE, NDistE, subs]];
NIntHLBb =
 Log[
  NIntegrate[
   SqrtDetEFlim[HLBb, DetHLBb, Nvals, Pvals, subs, FminMult, FmaxMult],
   ParmRange[[1]],
   ParmRange[[2]],
   ParmRange[[3]],
   AccuracyGoal → accgoal,
   PrecisionGoal → precgoal,
   Method → NIntMethod,
   MinRecursion → minRec,
   MaxRecursion → maxRec]]
,
Model == "MH",
DetMH = Det[EFisher[fisher[lLMH, {a, b, c}], MH, PDistE, NDistE, subs]];
NIntMH =
 Log[
  NIntegrate[
   SqrtDetEFlim[MH, DetMH, Nvals, Pvals, subs, FminMult, FmaxMult],
   ParmRange[[1]],
   ParmRange[[2]],
   ParmRange[[3]],
   AccuracyGoal → accgoal,
   PrecisionGoal → precgoal,
   Method → NIntMethod,
   MinRecursion → minRec,
   MaxRecursion → maxRec]]
,
Model == "MHb",
DetMHb = Det[EFisher[fisher[lLMHb, {a, b, c}], MHb, PDistE, NDistE, subs]];
NIntMHb =
 Log[
  NIntegrate[
```

```
    SqrtDetEFlim[MHb, DetMHb, Nvals, Pvals, subs, FminMult, FmaxMult],
    ParmRange[[1]],
    ParmRange[[2]],
    ParmRange[[3]],
    AccuracyGoal → accgoal,
    PrecisionGoal → precgoal,
    Method → NIntMethod,
    MinRecursion → minRec,
    MaxRecursion → maxRec]]
,
Model == "MHc",
DetMHc = Det[EFisher[fisher[lLMHc, {a, b, c}], MHc, PDistE, NDistE, subs]];
NIntMHc =
 Log[
  NIntegrate[
   SqrtDetEFlim[MHc, DetMHc, Nvals, Pvals, subs, FminMult, FmaxMult],
   ParmRange[[1]],
   ParmRange[[2]],
   ParmRange[[3]],
   AccuracyGoal → accgoal,
   PrecisionGoal → precgoal,
   Method → NIntMethod,
   MinRecursion → minRec,
   MaxRecursion → maxRec]]
,
Model == "To",
DetTo = Det[EFisher[fisher[lLTo, {a, b, c}], To, PDistE, NDistE, subs]];
NIntTo =
 Log[
  NIntegrate[
   SqrtDetEFlim[To, DetTo, Nvals, Pvals, subs, FminMult, FmaxMult],
   ParmRange[[1]],
   ParmRange[[2]],
   ParmRange[[3]],
   AccuracyGoal → accgoal,
   PrecisionGoal → precgoal,
   Method → NIntMethod,
   MinRecursion → minRec,
   MaxRecursion → maxRec]]
,
Model == "FHM",
DetFHM = Det[EFisher[fisher[lLFHM, {a, b, d}], FHM, PDistE, NDistE, subs]];
NIntFHM =
 Log[
```

```
  NIntegrate[
   SqrtDetEFlim[FHM, DetFHM, Nvals, Pvals, subs, FminMult, FmaxMult],
   ParmRange[[1]],
   ParmRange[[2]],
   ParmRange[[6]],
   AccuracyGoal → accgoal,
   PrecisionGoal → precgoal,
   Method → NIntMethod,
   MinRecursion → minRec,
   MaxRecursion → maxRec]]
,
Model == "W",
DetW = Det[EFisher[fisher[lLW, {a, b, v}], W, PDistE, NDistE, subs]];
NIntW =
 Log[
  NIntegrate[
   SqrtDetEFlim[W, DetW, Nvals, Pvals, subs, FminMult, FmaxMult],
   ParmRange[[1]],
   ParmRange[[2]],
   ParmRange[[4]],
   AccuracyGoal → accgoal,
   PrecisionGoal → precgoal,
   Method → NIntMethod,
   MinRecursion → minRec,
   MaxRecursion → maxRec]]
,
Model == "TTA",
DetTTA = Det[EFisher[fisher[lLTTA, {a, b, c}], TTA, PDistE, NDistE, subs]];
NIntTTA =
 Log[
  NIntegrate[
   SqrtDetEFlim[TTA, DetTTA, Nvals, Pvals, subs, FminMult, FmaxMult],
   ParmRange[[1]],
   ParmRange[[2]],
   ParmRange[[3]],
   AccuracyGoal → accgoal,
   PrecisionGoal → precgoal,
   Method → NIntMethod,
   MinRecursion → minRec,
   MaxRecursion → maxRec]]
,
Model == "SBB",
DetSBB = Det[EFisher[fisher[lLSBB, {a, b, v}], SBB, PDistE, NDistE, subs]];
NIntSBB =
```

```
 Log[
  NIntegrate[
   SqrtDetEFlim[SBB, DetSBB, Nvals, Pvals, subs, FminMult, FmaxMult],
   ParmRange[[1]],
   ParmRange[[2]],
   ParmRange[[4]],
   AccuracyGoal → accgoal,
   PrecisionGoal → precgoal,
   Method → NIntMethod,
   MinRecursion → minRec,
   MaxRecursion → maxRec]]
,
Model == "SSS",
DetSSS = Det[EFisher[fisher[lLSSS, {a, b, c}], SSS, PDistE, NDistE, subs]];
NIntSSS =
 Log[
  NIntegrate[
   SqrtDetEFlim[SSS, DetSSS, Nvals, Pvals, subs, FminMult, FmaxMult],
   ParmRange[[1]],
   ParmRange[[2]],
   ParmRange[[3]],
   AccuracyGoal → accgoal,
   PrecisionGoal → precgoal,
   Method → NIntMethod,
   MinRecursion → minRec,
   MaxRecursion → maxRec]]
,
Model == "RGD",
DetRGD = Det[EFisher[fisher[lLRGD, {a, b, c}], RGD, PDistE, NDistE, subs]];
NIntRGD =
 Log[
  NIntegrate[
   SqrtDetEFlim[RGD, DetRGD, Nvals, Pvals, subs, FminMult, FmaxMult],
   ParmRange[[1]],
   ParmRange[[2]],
   ParmRange[[3]],
   AccuracyGoal → accgoal,
   PrecisionGoal → precgoal,
   Method → NIntMethod,
   MinRecursion → minRec,
   MaxRecursion → maxRec]]
,
Model == "S3",
DetS3 = Det[EFisher[fisher[lLS3, {a, b, u}], S3, PDistE, NDistE, subs]];
```

```
NIntS3 =
 Log[
  NIntegrate[
   SqrtDetEFlim[S3, DetS3, Nvals, Pvals, subs, FminMult, FmaxMult],
   ParmRange[[1]],
   ParmRange[[2]],
   ParmRange[[5]],
   AccuracyGoal → accgoal,
   PrecisionGoal → precgoal,
   Method → NIntMethod,
   MinRecursion → minRec,
   MaxRecursion → maxRec]]
,
Model == "S3b",
DetS3b = Det[EFisher[fisher[lLS3b, {a, b, u}], S3b, PDistE, NDistE, subs]];
NIntS3b =
 Log[
  NIntegrate[
   SqrtDetEFlim[S3b, DetS3b, Nvals, Pvals, subs, FminMult, FmaxMult],
   ParmRange[[1]],
   ParmRange[[2]],
   ParmRange[[5]],
   AccuracyGoal → accgoal,
   PrecisionGoal → precgoal,
   Method → NIntMethod,
   MinRecursion → minRec,
   MaxRecursion → maxRec]]
,
Model == "BDOR",
DetBDOR = Det[EFisher[fisher[lLBDOR, {a, b, c, u}], BDOR, PDistE, NDistE, subs]];
NIntBDOR =
 Log[
  NIntegrate[
   SqrtDetEFlim[BDOR, DetBDOR, Nvals, Pvals, subs, FminMult, FmaxMult],
   ParmRange[[1]],
   ParmRange[[2]],
   ParmRange[[3]],
   ParmRange[[5]],
   AccuracyGoal → accgoal,
   PrecisionGoal → precgoal,
   Method → NIntMethod,
   MinRecursion → 300,
   MaxRecursion → 1000]]
,
```

```
Model == "CMOR",
DetCMOR = Det[EFisher[fisher[lLCMOR, {a, b, c, u}], CMOR, PDistE, NDistE, subs]];
NIntCMOR =
 Log[
  NIntegrate[
   SqrtDetEFlim[CMOR, DetCMOR, Nvals, Pvals, subs, FminMult, FmaxMult],
   ParmRange[[1]],
   ParmRange[[2]],
   ParmRange[[3]],
   ParmRange[[5]],
   AccuracyGoal → accgoal,
   PrecisionGoal → precgoal,
   Method → NIntMethod,
   MinRecursion → 300,
   MaxRecursion → 1000]]
,
Model == "AAOR",
DetAAOR = Det[EFisher[fisher[lLAAOR, {a, b, v, u}], AAOR, PDistE, NDistE, subs]];
NIntAAOR =
 Log[
  NIntegrate[
   SqrtDetEFlim[AAOR, DetAAOR, Nvals, Pvals, subs, FminMult, FmaxMult],
   ParmRange[[1]],
   ParmRange[[2]],
   ParmRange[[4]],
   ParmRange[[5]],
   AccuracyGoal → accgoal,
   PrecisionGoal → precgoal,
   Method → NIntMethod,
   MinRecursion → 300,
   MaxRecursion → 1000]]
,
Model == "SN1",
DetSN1 = Det[EFisher[fisher[lLSN1, {a, b, c, d}], SN1, PDistE, NDistE, subs]];
NIntSN1 =
 Log[
  NIntegrate[
   SqrtDetEFlimSN1[SN1, DetSN1, Nvals, Pvals, subs, FminMult, FmaxMult],
   ParmRange[[1]],
   ParmRange[[2]],
   ParmRange[[3]],
   ParmRange[[6]],
   AccuracyGoal → accgoal,
   PrecisionGoal → precgoal,
```

```
      Method → NIntMethod,
      MinRecursion → 300,
      MaxRecursion → 1000]]
  ,
  Model == "SN2",
  DetSN2 = Det[EFisher[
      fisher[lLSN2, {a, b, c, d}], SN2, PDistE, NDistE, subs]];
  NIntSN2 =
   Log[
    NIntegrate[
     SqrtDetEFlimSN2[SN2, DetSN2, Nvals, Pvals, subs, FminMult, FmaxMult],
     ParmRange[[1]],
     ParmRange[[2]],
     ParmRange[[3]],
     ParmRange[[6]],
     AccuracyGoal → accgoal,
     PrecisionGoal → precgoal,
     Method → NIntMethod,
     MinRecursion → 300,
     MaxRecursion → 1000]]
  ]
 ]
```

Define functions to create experimental designs

The “GoldenRatio” series of abundance levels (starting at 3 for prey and 1 for predators) enabling the generation of arbitrary numbers of equidistant points between min and (specified) max abundances.
The series will be approximately Fibonacci when PreyMax is set to *Fibonacci[n]* for a desired series length *n*.
The “Arithmetic series” spaces out levels equidistantly in arithmetic space to the same maximum as used for the “GoldenRatio” series.

In[•]:=
```
logGRSpace[a_, b_, n_] := Round[GoldenRatio^Range[a, b, (b - a) / (n - 1)]]

PreyVals[n_, PreyMax_, AbundSeries_] :=
 Which[
  AbundSeries == "GoldenRatio",
  logGRSpace[2, Log[GoldenRatio, PreyMax] + 3, n],
  AbundSeries == "Arithmetic",
  Round[Range[3, Max[logGRSpace[2, Log[GoldenRatio, PreyMax] + 3, n]],
    (Max[logGRSpace[2, Log[GoldenRatio, PreyMax] + 3, n]] - 3) / (n - 1)]]
 ]
PredVals[n_, PredMax_, AbundSeries_] :=
 If[n == 1,
  {1}, (* If only a single level is requested,
  specify a single predator individual *)
  Which[ (* Otherwise, determine predator levels according to GoldenRatio
    or Arithmetic series beginning with 1 predator individual *)
   AbundSeries == "GoldenRatio",
   logGRSpace[0, Log[GoldenRatio, PredMax] + 1, n],
   AbundSeries == "Arithmetic",
   Round[Range[1, Max[logGRSpace[0, Log[GoldenRatio, PredMax] + 1, n]],
     (Max[logGRSpace[0, Log[GoldenRatio, PredMax] + 1, n]] - 1) / (n - 1)]]
  ]]
```

Specify AbundanceSeries and determine and export designs

Be sure to copy numbers to GeomComp[] function below.

In[•]:=
```
PreyMinLevels = 5;
PreyMaxLevelsVar = 10;
PreyMaxLevelsFix = 10;
PredMinLevels = 1;
PredMaxLevelsVar = 5;
PredMaxLevelsFix = 4;
```

Export "Var" designs - increasing length and increasing maximum value

In[•]:=
```
VarDesigns = Table[
    {PreyVals[i, Fibonacci[i], AbundSeries],
     PredVals[j, Fibonacci[j], AbundSeries]},
    {i, PreyMinLevels, PreyMaxLevelsVar},
    {j, PredMinLevels, PredMaxLevelsVar}
   ];
TableForm[VarDesigns];
Dimensions[VarDesigns]
(*
Export["DesignsVar1.txt",TeXForm[VarDesigns[[All,1;;3,All]]]];
Export["DesignsVar2.txt",TeXForm[VarDesigns[[All,4;;5,All]]]];
*)
```

Out[•]= {6, 5, 2}

Export "Fix" designs - varying length and constant maximum value

In[•]:=
```
FixDesigns = Table[
    {PreyVals[i, Fibonacci[PreyMaxLevelsFix], AbundSeries],
     PredVals[j, Fibonacci[PredMaxLevelsFix], AbundSeries]},
    {i, PreyMinLevels, PreyMaxLevelsFix},
    {j, PredMinLevels, PredMaxLevelsFix}
   ];
TableForm[FixDesigns];
Dimensions[FixDesigns]
(*
Export["DesignsFix1.txt",TeXForm[FixDesigns[[All,1;;2,All]]]];
Export["DesignsFix2.txt",TeXForm[FixDesigns[[All,3;;4,All]]]];
*)
```

Out[•]= {6, 4, 2}

Define “GeomComp[]” Wrapper to apply “GeomComplex[]” across experimental designs

In[•]:=
```
ClearAll[GeomComp];
GeomComp[
  ModelAbb_,
  Type_,
  AbundSeries_] :=
 Module[
  {
```

```
 (******* Be sure to match the following with exported designs above *******)
 PreyMinLevels = 5,
 PreyMaxLevelsVar = 10,
 PreyMaxLevelsFix = 10,
 PredMinLevels = 1,
 PredMaxLevelsVar = 5,
 PredMaxLevelsFix = 4
}
,
Which[
 Type == "Var",
 Flatten[
  ParallelTable[
   Flatten[{
     Max[PreyVals[i, Fibonacci[i], AbundSeries]], (* Maximum prey level *)
     Max[PredVals[j, Fibonacci[j], AbundSeries]], (* Maximum pred level *)
     Length[PreyVals[i, Fibonacci[i], AbundSeries]],
     (* Number of prey levels *)
     Length[PredVals[j, Fibonacci[j], AbundSeries]],
     (* Number of pred levels *)
     GeomComplex[
      PreyVals[i, Fibonacci[i], AbundSeries],
      PredVals[j, Fibonacci[j], AbundSeries],
      Model = ModelAbb]
    }],
   {i, PreyMinLevels, PreyMaxLevelsVar},
   {j, PredMinLevels, PredMaxLevelsVar}
  ],
  1]
 ,
 Type == "Fix",
 Flatten[
  ParallelTable[
   Flatten[{
     Max[PreyVals[i, Fibonacci[PreyMaxLevelsFix], AbundSeries]],
     (* Maximum prey level *)
     Max[PredVals[j, Fibonacci[PredMaxLevelsFix], AbundSeries]],
     (* Maximum pred level *)
     Length[PreyVals[i, Fibonacci[PreyMaxLevelsFix], AbundSeries]],
     (* Number of prey levels *)
     Length[PredVals[j, Fibonacci[PredMaxLevelsFix], AbundSeries]],
     (* Number of pred levels *)
     GeomComplex[
      PreyVals[i, Fibonacci[PreyMaxLevelsFix], AbundSeries],
```

```
        PredVals[j, Fibonacci[PredMaxLevelsFix], AbundSeries],
        Model = ModelAbb]
      }],
     {i, PreyMinLevels, PreyMaxLevelsFix},
     {j, PredMinLevels, PredMaxLevelsFix}
    ],
    1]
  ]
 ]
```

Apply across designs

k = 1 models

In[•]:=
```
varH1 = GeomComp[Model = "H1", Type = "Var", AbundSeries = AbundSeries];
DeleteFile["k1varH1.txt"]; Save["k1varH1.txt", varH1]
```

In[•]:=

In[•]:=
```
fixH1 = GeomComp[Model = "H1", Type = "Fix", AbundSeries = AbundSeries];
DeleteFile["k1fixH1.txt"]; Save["k1fixH1.txt", fixH1]
```

In[•]:=

In[•]:=
```
varLR = GeomComp[Model = "LR", Type = "Var", AbundSeries = AbundSeries];
DeleteFile["k1varLR.txt"]; Save["k1varLR.txt", varLR]
```

In[•]:=
```
fixLR = GeomComp[Model = "LR", Type = "Fix", AbundSeries = AbundSeries];
DeleteFile["k1fixLR.txt"]; Save["k1fixLR.txt", fixLR]
```

In[•]:=

In[•]:=
```
varBWL1 = GeomComp[Model = "BWL1", Type = "Var", AbundSeries = AbundSeries];
DeleteFile["k1varBWL1.txt"]; Save["k1varBWL1.txt", varBWL1]
```

In[•]:=

In[•]:=
```
fixBWL1 = GeomComp[Model = "BWL1", Type = "Fix", AbundSeries = AbundSeries];
DeleteFile["k1fixBWL1.txt"]; Save["k1fixBWL1.txt", fixBWL1]
```

In[•]:=

k = 2 models

In[•]:=
```
varH2 = GeomComp[Model = "H2", Type = "Var", AbundSeries = AbundSeries];
DeleteFile["k2varH2.txt"]; Save["k2varH2.txt", varH2]
```

Out[•]=
```
$Aborted
```

In[•]:=
```
fixH2 = GeomComp[Model = "H2", Type = "Fix", AbundSeries = AbundSeries];
DeleteFile["k2fixH2.txt"]; Save["k2fixH2.txt", fixH2]
```

--- DeleteFile: Directory or file k2fixH2.txt not found.

In[•]:=

In[•]:=
```
varMM = GeomComp[Model = "MM", Type = "Var", AbundSeries = AbundSeries];
DeleteFile["k2varMM.txt"]; Save["k2varMM.txt", varMM]
```

--- DeleteFile: Directory or file k2varMM.txt not found.

In[•]:=
```
fixMM = GeomComp[Model = "MM", Type = "Fix", AbundSeries = AbundSeries];
DeleteFile["k2fixMM.txt"]; Save["k2fixMM.txt", fixMM]
```

--- DeleteFile: Directory or file k2fixMM.txt not found.

In[•]:=

In[•]:=
```
varHV = GeomComp[Model = "HV", Type = "Var", AbundSeries = AbundSeries];
DeleteFile["k2varHV.txt"]; Save["k2varHV.txt", varHV]
```

--- DeleteFile: Directory or file k2varHV.txt not found.

In[•]:=
```
fixHV = GeomComp[Model = "HV", Type = "Fix", AbundSeries = AbundSeries];
DeleteFile["k2fixHV.txt"]; Save["k2fixHV.txt", fixHV]
```

--- DeleteFile: Directory or file k2fixHV.txt not found.

In[•]:=

In[•]:=
```
varR = GeomComp[Model = "R", Type = "Var", AbundSeries = AbundSeries];
DeleteFile["k2varR.txt"]; Save["k2varR.txt", varR]
```

--- DeleteFile: Directory or file k2varR.txt not found.

In[•]:=
```
fixR = GeomComp[Model = "R", Type = "Fix", AbundSeries = AbundSeries];
DeleteFile["k2fixR.txt"]; Save["k2fixR.txt", fixR]
```

--- DeleteFile: Directory or file k2fixR.txt not found.

In[•]:=

In[•]:=
```
varAG = GeomComp[Model = "AG", Type = "Var", AbundSeries = AbundSeries];
DeleteFile["k2varAG.txt"]; Save["k2varAG.txt", varAG]
```

--- DeleteFile: Directory or file k2varAG.txt not found.

In[•]:=
```
fixAG = GeomComp[Model = "AG", Type = "Fix", AbundSeries = AbundSeries];
DeleteFile["k2fixAG.txt"]; Save["k2fixAG.txt", fixAG]
```

--- DeleteFile: Directory or file k2fixAG.txt not found.

In[•]:=

In[•]:=
```
varCDAO = GeomComp[Model = "CDAO", Type = "Var", AbundSeries = AbundSeries];
DeleteFile["k2varCDAO.txt"]; Save["k2varCDAO.txt", varCDAO]
```

--- DeleteFile: Directory or file k2varCDAO.txt not found.

In[•]:=
```
fixCDAO = GeomComp[Model = "CDAO", Type = "Fix", AbundSeries = AbundSeries];
DeleteFile["k2fixCDAO.txt"]; Save["k2fixCDAO.txt", fixCDAO]
```

--- DeleteFile: Directory or file k2fixCDAO.txt not found.

In[•]:=

In[•]:=
```
varGI = GeomComp[Model = "GI", Type = "Var", AbundSeries = AbundSeries];
DeleteFile["k2varGI.txt"]; Save["k2varGI.txt", varGI]
```

--- DeleteFile: Directory or file k2varGI.txt not found.

In[•]:=
```
fixGI = GeomComp[Model = "GI", Type = "Fix", AbundSeries = AbundSeries];
DeleteFile["k2fixGI.txt"]; Save["k2fixGI.txt", fixGI]
```

--- DeleteFile: Directory or file k2fixGI.txt not found.

In[•]:=

In[•]:=
```
varGIA = GeomComp[Model = "GIA", Type = "Var", AbundSeries = AbundSeries];
DeleteFile["k2varGIA.txt"]; Save["k2varGIA.txt", varGIA]
```

--- DeleteFile: Directory or file k2varGIA.txt not found.

In[•]:=
```
fixGIA = GeomComp[Model = "GIA", Type = "Fix", AbundSeries = AbundSeries];
DeleteFile["k2fixGIA.txt"]; Save["k2fixGIA.txt", fixGIA]
```

--- DeleteFile: Directory or file k2fixGIA.txt not found.

In[•]:=

In[•]:=
```
varGB = GeomComp[Model = "GB", Type = "Var", AbundSeries = AbundSeries];
DeleteFile["k2varGB.txt"]; Save["k2varGB.txt", varGB]
```

--- DeleteFile: Directory or file k2varGB.txt not found.

In[•]:=
```
fixGB = GeomComp[Model = "GB", Type = "Fix", AbundSeries = AbundSeries];
DeleteFile["k2fixGB.txt"]; Save["k2fixGB.txt", fixGB]
```

--- DeleteFile: Directory or file k2fixGB.txt not found.

In[•]:=

In[•]:=
```
varHT = GeomComp[Model = "HT", Type = "Var", AbundSeries = AbundSeries];
DeleteFile["k2varHT.txt"];
Save["k2varHT.txt", varHT]
```

--- DeleteFile: Directory or file k2varHT.txt not found.

In[•]:=
```
fixHT = GeomComp[Model = "HT", Type = "Fix", AbundSeries = AbundSeries];
DeleteFile["k2fixHT.txt"]; Save["k2fixHT.txt", fixHT]
```

--- DeleteFile: Directory or file k2fixHT.txt not found.

In[•]:=

In[•]:=
```
varHTb = GeomComp[Model = "HTb", Type = "Var", AbundSeries = AbundSeries];
DeleteFile["k2varHTb.txt"];
Save["k2varHTb.txt", varHTb]
```

--- DeleteFile: Directory or file k2varHTb.txt not found.

In[•]:=
```
fixHTb = GeomComp[Model = "HTb", Type = "Fix", AbundSeries = AbundSeries];
DeleteFile["k2fixHTb.txt"]; Save["k2fixHTb.txt", fixHTb]
```

--- DeleteFile: Directory or file k2fixHTb.txt not found.

In[•]:=

In[•]:=
```
varH3 = GeomComp[Model = "H3", Type = "Var", AbundSeries = AbundSeries];
DeleteFile["k2varH3.txt"]; Save["k2varH3.txt", varH3]
```

--- DeleteFile: Directory or file k2varH3.txt not found.

In[•]:=
```
fixH3 = GeomComp[Model = "H3", Type = "Fix", AbundSeries = AbundSeries];
DeleteFile["k2fixH3.txt"]; Save["k2fixH3.txt", fixH3]
```

--- DeleteFile: Directory or file k2fixH3.txt not found.

In[•]:=

In[•]:=
```
varAGK = GeomComp[Model = "AGK", Type = "Var", AbundSeries = AbundSeries];
DeleteFile["k2varAGK.txt"]; Save["k2varAGK.txt", varAGK]
```

--- DeleteFile: Directory or file k2varAGK.txt not found.

In[•]:=
```
fixAGK = GeomComp[Model = "AGK", Type = "Fix", AbundSeries = AbundSeries];
DeleteFile["k2fixAGK.txt"]; Save["k2fixAGK.txt", fixAGK]
```

--- DeleteFile: Directory or file k2fixAGK.txt not found.

In[•]:=

In[•]:=
```
varA0 = GeomComp[Model = "A0", Type = "Var", AbundSeries = AbundSeries];
DeleteFile["k2varA0.txt"]; Save["k2varA0.txt", varA0]
```
--- DeleteFile: Directory or file k2varA0.txt not found.

In[•]:=
```
fixA0 = GeomComp[Model = "A0", Type = "Fix", AbundSeries = AbundSeries];
DeleteFile["k2fixA0.txt"]; Save["k2fixA0.txt", fixA0]
```
--- DeleteFile: Directory or file k2fixA0.txt not found.

In[•]:=

In[•]:=
```
varA1 = GeomComp[Model = "A1", Type = "Var", AbundSeries = AbundSeries];
DeleteFile["k2varA1.txt"]; Save["k2varA1.txt", varA1]
```
--- DeleteFile: Directory or file k2varA1.txt not found.

In[•]:=
```
fixA1 = GeomComp[Model = "A1", Type = "Fix", AbundSeries = AbundSeries];
DeleteFile["k2fixA1.txt"]; Save["k2fixA1.txt", fixA1]
```
--- DeleteFile: Directory or file k2fixA1.txt not found.

In[•]:=

In[•]:=
```
varA3 = GeomComp[Model = "A3", Type = "Var", AbundSeries = AbundSeries];
DeleteFile["k2varA3.txt"]; Save["k2varA3.txt", varA3]
```
--- DeleteFile: Directory or file k2varA3.txt not found.

In[•]:=
```
fixA3 = GeomComp[Model = "A3", Type = "Fix", AbundSeries = AbundSeries];
DeleteFile["k2fixA3.txt"]; Save["k2fixA3.txt", fixA3]
```
--- DeleteFile: Directory or file k2fixA3.txt not found.

In[•]:=

In[•]:=
```
varSH = GeomComp[Model = "SH", Type = "Var", AbundSeries = AbundSeries];
DeleteFile["k2varSH.txt"]; Save["k2varSH.txt", varSH]
```
--- DeleteFile: Directory or file k2varSH.txt not found.

In[•]:=
```
fixSH = GeomComp[Model = "SH", Type = "Fix", AbundSeries = AbundSeries];
DeleteFile["k2fixSH.txt"]; Save["k2fixSH.txt", fixSH]
```
--- DeleteFile: Directory or file k2fixSH.txt not found.

In[•]:=

In[•]:=
```
varSHb = GeomComp[Model = "SHb", Type = "Var", AbundSeries = AbundSeries];
DeleteFile["k2varSHb.txt"]; Save["k2varSHb.txt", varSHb]
```

--- DeleteFile: Directory or file k2varSHb.txt not found.

In[•]:=
```
fixSHb = GeomComp[Model = "SHb", Type = "Fix", AbundSeries = AbundSeries];
DeleteFile["k2fixSHb.txt"]; Save["k2fixSHb.txt", fixSHb]
```

--- DeleteFile: Directory or file k2fixSHb.txt not found.

In[•]:=

k = 3 models

In[•]:=
```
varBD = GeomComp[Model = "BD", Type = "Var", AbundSeries = AbundSeries];
DeleteFile["k3varBD.txt"]; Save["k3varBD.txt", varBD]
```

--- DeleteFile: Directory or file k3varBD.txt not found.

In[•]:=
```
fixBD = GeomComp[Model = "BD", Type = "Fix", AbundSeries = AbundSeries];
DeleteFile["k3fixBD.txt"]; Save["k3fixBD.txt", fixBD]
```

--- DeleteFile: Directory or file k3fixBD.txt not found.

In[•]:=

In[•]:=
```
varCM = GeomComp[Model = "CM", Type = "Var", AbundSeries = AbundSeries];
DeleteFile["k3varCM.txt"]; Save["k3varCM.txt", varCM]
```

--- DeleteFile: Directory or file k3varCM.txt not found.

In[•]:=
```
fixCM = GeomComp[Model = "CM", Type = "Fix", AbundSeries = AbundSeries];
DeleteFile["k3fixCM.txt"]; Save["k3fixCM.txt", fixCM]
```

--- DeleteFile: Directory or file k3fixCM.txt not found.

In[•]:=

In[•]:=
```
varAA = GeomComp[Model = "AA", Type = "Var", AbundSeries = AbundSeries];
DeleteFile["k3varAA.txt"]; Save["k3varAA.txt", varAA]
```

--- DeleteFile: Directory or file k3varAA.txt not found.

In[•]:=
```
fixAA = GeomComp[Model = "AA", Type = "Fix", AbundSeries = AbundSeries];
DeleteFile["k3fixAA.txt"]; Save["k3fixAA.txt", fixAA]
```

--- DeleteFile: Directory or file k3fixAA.txt not found.

In[•]:=

In[•]:=
```
varBWL2 = GeomComp[Model = "BWL2", Type = "Var", AbundSeries = AbundSeries];
DeleteFile["k3varBWL2.txt"]; Save["k3varBWL2.txt", varBWL2]
```

In[•]:=
```
fixBWL2 = GeomComp[Model = "BWL2", Type = "Fix", AbundSeries = AbundSeries];
DeleteFile["k3fixBWL2.txt"]; Save["k3fixBWL2.txt", fixBWL2]
```

In[•]:=

In[•]:=
```
varH3R = GeomComp[Model = "H3R", Type = "Var", AbundSeries = AbundSeries];
DeleteFile["k3varH3R.txt"]; Save["k3varH3R.txt", varH3R]
```

--- DeleteFile: Directory or file k3varH3R.txt not found.

In[•]:=
```
fixH3R = GeomComp[Model = "H3R", Type = "Fix", AbundSeries = AbundSeries];
DeleteFile["k3fixH3R.txt"]; Save["k3fixH3R.txt", fixH3R]
```

--- DeleteFile: Directory or file k3fixH3R.txt not found.

In[•]:=

In[•]:=
```
varA2 = GeomComp[Model = "A2", Type = "Var", AbundSeries = AbundSeries];
DeleteFile["k3varA2.txt"]; Save["k3varA2.txt", varA2]
```

--- DeleteFile: Directory or file k3varA2.txt not found.

In[•]:=
```
fixA2 = GeomComp[Model = "A2", Type = "Fix", AbundSeries = AbundSeries];
DeleteFile["k3fixA2.txt"]; Save["k3fixA2.txt", fixA2]
```

--- DeleteFile: Directory or file k3fixA2.txt not found.

In[•]:=

In[•]:=
```
varHLB = GeomComp[Model = "HLB", Type = "Var", AbundSeries = AbundSeries];
DeleteFile["k3varHLB.txt"]; Save["k3varHLB.txt", varHLB]
```

--- DeleteFile: Directory or file k3varHLB.txt not found.

In[•]:=
```
fixHLB = GeomComp[Model = "HLB", Type = "Fix", AbundSeries = AbundSeries];
DeleteFile["k3fixHLB.txt"]; Save["k3fixHLB.txt", fixHLB]
```

--- DeleteFile: Directory or file k3fixHLB.txt not found.

In[•]:=

In[•]:=
```
varHLBb = GeomComp[Model = "HLBb", Type = "Var", AbundSeries = AbundSeries];
DeleteFile["k3varHLBb.txt"]; Save["k3varHLBb.txt", varHLBb]
```

--- DeleteFile: Directory or file k3varHLBb.txt not found.

In[•]:=
```
fixHLBb = GeomComp[Model = "HLBb", Type = "Fix", AbundSeries = AbundSeries];
DeleteFile["k3fixHLBb.txt"]; Save["k3fixHLBb.txt", fixHLBb]
```

--- DeleteFile: Directory or file k3fixHLBb.txt not found.

In[•]:=

In[•]:=
```
varMH = GeomComp[Model = "MH", Type = "Var", AbundSeries = AbundSeries];
DeleteFile["k3varMH.txt"]; Save["k3varMH.txt", varMH]
```

--- DeleteFile: Directory or file k3varMH.txt not found.

In[•]:=
```
fixMH = GeomComp[Model = "MH", Type = "Fix", AbundSeries = AbundSeries];
DeleteFile["k3fixMH.txt"]; Save["k3fixMH.txt", fixMH]
```

--- DeleteFile: Directory or file k3fixMH.txt not found.

In[•]:=

In[•]:=
```
varMHb = GeomComp[Model = "MHb", Type = "Var", AbundSeries = AbundSeries];
DeleteFile["k3varMHb.txt"]; Save["k3varMHb.txt", varMHb]
```

--- DeleteFile: Directory or file k3varMHb.txt not found.

In[•]:=
```
fixMHb = GeomComp[Model = "MHb", Type = "Fix", AbundSeries = AbundSeries];
DeleteFile["k3fixMHb.txt"]; Save["k3fixMHb.txt", fixMHb]
```

--- DeleteFile: Directory or file k3fixMHb.txt not found.

In[•]:=

In[•]:=
```
varMHc = GeomComp[Model = "MHc", Type = "Var", AbundSeries = AbundSeries];
DeleteFile["k3varMHc.txt"]; Save["k3varMHc.txt", varMHc]
```

--- DeleteFile: Directory or file k3varMHc.txt not found.

In[•]:=
```
fixMHc = GeomComp[Model = "MHc", Type = "Fix", AbundSeries = AbundSeries];
DeleteFile["k3fixMHc.txt"]; Save["k3fixMHc.txt", fixMHc]
```

--- DeleteFile: Directory or file k3fixMHc.txt not found.

In[•]:=

In[•]:=
```
varT = GeomComp[Model = "To", Type = "Var", AbundSeries = AbundSeries];
DeleteFile["k3varT.txt"]; Save["k3varT.txt", varT]
```

--- DeleteFile: Directory or file k3varT.txt not found.

In[•]:=
```
fixT = GeomComp[Model = "To", Type = "Fix", AbundSeries = AbundSeries];
DeleteFile["k3fixT.txt"]; Save["k3fixT.txt", fixT]
```

--- DeleteFile: Directory or file k3fixT.txt not found.

In[•]:=

In[•]:=
```
varFHM = GeomComp[Model = "FHM", Type = "Var", AbundSeries = AbundSeries];
DeleteFile["k3varFHM.txt"]; Save["k3varFHM.txt", varFHM]
```

--- DeleteFile: Directory or file k3varFHM.txt not found.

In[ ]:=
```
fixFHM = GeomComp[Model = "FHM", Type = "Fix", AbundSeries = AbundSeries];
DeleteFile["k3fixFHM.txt"]; Save["k3fixFHM.txt", fixFHM]
```

--- DeleteFile: Directory or file k3fixFHM.txt not found.

In[ ]:=

In[ ]:=
```
varW = GeomComp[Model = "W", Type = "Var", AbundSeries = AbundSeries];
DeleteFile["k3varW.txt"]; Save["k3varW.txt", varW]
```

--- DeleteFile: Directory or file k3varW.txt not found.

In[ ]:=
```
fixW = GeomComp[Model = "W", Type = "Fix", AbundSeries = AbundSeries];
DeleteFile["k3fixW.txt"]; Save["k3fixW.txt", fixW]
```

--- DeleteFile: Directory or file k3fixW.txt not found.

In[ ]:=

In[ ]:=
```
varTTA = GeomComp[Model = "TTA", Type = "Var", AbundSeries = AbundSeries];
DeleteFile["k3varTTA.txt"]; Save["k3varTTA.txt", varTTA]
```

--- DeleteFile: Directory or file k3varTTA.txt not found.

In[ ]:=
```
fixTTA = GeomComp[Model = "TTA", Type = "Fix", AbundSeries = AbundSeries];
DeleteFile["k3fixTTA.txt"]; Save["k3fixTTA.txt", fixTTA]
```

--- DeleteFile: Directory or file k3fixTTA.txt not found.

In[ ]:=

In[ ]:=
```
varSBB = GeomComp[Model = "SBB", Type = "Var", AbundSeries = AbundSeries];
DeleteFile["k3varSBB.txt"]; Save["k3varSBB.txt", varSBB]
```

--- DeleteFile: Directory or file k3varSBB.txt not found.

In[ ]:=
```
fixSBB = GeomComp[Model = "SBB", Type = "Fix", AbundSeries = AbundSeries];
DeleteFile["k3fixSBB.txt"]; Save["k3fixSBB.txt", fixSBB]
```

--- DeleteFile: Directory or file k3fixSBB.txt not found.

In[ ]:=

In[ ]:=
```
varSSS = GeomComp[Model = "SSS", Type = "Var", AbundSeries = AbundSeries];
DeleteFile["k3varSSS.txt"];
Save["k3varSSS.txt", varSSS]
```

--- DeleteFile: Directory or file k3varSSS.txt not found.

In[•]:=
```
fixSSS = GeomComp[Model = "SSS", Type = "Fix", AbundSeries = AbundSeries];
DeleteFile["k3fixSSS.txt"];
Save["k3fixSSS.txt", fixSSS]
```

--- DeleteFile: Directory or file k3fixSSS.txt not found.

In[•]:=

In[•]:=
```
varRGD = GeomComp[Model = "RGD", Type = "Var", AbundSeries = AbundSeries];
DeleteFile["k3varRGD.txt"];
Save["k3varRGD.txt", varRGD]
```

In[•]:=
```
fixRGD = GeomComp[Model = "RGD", Type = "Fix", AbundSeries = AbundSeries];
DeleteFile["k3fixRGD.txt"];
Save["k3fixRGD.txt", fixRGD]
```

In[•]:=

In[•]:=
```
varS3 = GeomComp[Model = "S3", Type = "Var", AbundSeries = AbundSeries];
DeleteFile["k3varS3.txt"];
Save["k3varS3.txt", varS3]
```

--- DeleteFile: Directory or file k3varS3.txt not found.

In[•]:=
```
fixS3 = GeomComp[Model = "S3", Type = "Fix", AbundSeries = AbundSeries];
DeleteFile["k3fixS3.txt"];
Save["k3fixS3.txt", fixS3]
```

--- DeleteFile: Directory or file k3fixS3.txt not found.

In[•]:=

In[•]:=
```
varS3b = GeomComp[Model = "S3b", Type = "Var", AbundSeries = AbundSeries];
DeleteFile["k3varS3b.txt"];
Save["k3varS3b.txt", varS3b]
```

--- DeleteFile: Directory or file k3varS3b.txt not found.

In[•]:=
```
fixS3b = GeomComp[Model = "S3b", Type = "Fix", AbundSeries = AbundSeries];
DeleteFile["k3fixS3b.txt"];
Save["k3fixS3b.txt", fixS3b]
```

--- DeleteFile: Directory or file k3fixS3b.txt not found.

In[•]:=

k = 4 models

In[•]:=
```
varBDOR = GeomComp[Model = "BDOR", Type = "Var", AbundSeries = AbundSeries];
DeleteFile["k4varBDOR.txt"]; Save["k4varBDOR.txt", varBDOR]
```

--- DeleteFile: Directory or file k4varBDOR.txt not found.

In[•]:=
```
fixBDOR = GeomComp[Model = "BDOR", Type = "Fix", AbundSeries = AbundSeries];
DeleteFile["k4fixBDOR.txt"]; Save["k4fixBDOR.txt", fixBDOR]
```

--- DeleteFile: Directory or file k4fixBDOR.txt not found.

In[•]:=

In[•]:=
```
varCMOR = GeomComp[Model = "CMOR", Type = "Var", AbundSeries = AbundSeries];
DeleteFile["k4varCMOR.txt"]; Save["k4varCMOR.txt", varCMOR]
```

--- DeleteFile: Directory or file k4varCMOR.txt not found.

In[•]:=
```
fixCMOR = GeomComp[Model = "CMOR", Type = "Fix", AbundSeries = AbundSeries];
DeleteFile["k4fixCMOR.txt"]; Save["k4fixCMOR.txt", fixCMOR]
```

--- DeleteFile: Directory or file k4fixCMOR.txt not found.

In[•]:=

In[•]:=
```
varAAOR = GeomComp[Model = "AAOR", Type = "Var", AbundSeries = AbundSeries];
DeleteFile["k4varAAOR.txt"]; Save["k4varAAOR.txt", varAAOR]
```

--- DeleteFile: Directory or file k4varAAOR.txt not found.

In[•]:=
```
fixAAOR = GeomComp[Model = "AAOR", Type = "Fix", AbundSeries = AbundSeries];
DeleteFile["k4fixAAOR.txt"]; Save["k4fixAAOR.txt", fixAAOR]
```

--- DeleteFile: Directory or file k4fixAAOR.txt not found.

In[•]:=

In[•]:=
```
varSN1 = GeomComp[Model = "SN1", Type = "Var", AbundSeries = AbundSeries];
DeleteFile["k4varSN1.txt"];
Save["k4varSN1.txt", varSN1]
```

--- DeleteFile: Directory or file k4varSN1.txt not found.

In[•]:=
```
fixSN1 = GeomComp[Model = "SN1", Type = "Fix", AbundSeries = AbundSeries];
DeleteFile["k4fixSN1.txt"];
Save["k4fixSN1.txt", fixSN1]
```

--- DeleteFile: Directory or file k4fixSN1.txt not found.

In[•]:=

In[•]:=
```
varSN2 = GeomComp[Model = "SN2", Type = "Var", AbundSeries = AbundSeries];
DeleteFile["k4varSN2.txt"];
Save["k4varSN2.txt", varSN2]
```

--- DeleteFile: Directory or file k4varSN2.txt not found.

In[•]:=
```
fixSN2 = GeomComp[Model = "SN2", Type = "Fix", AbundSeries = AbundSeries];
DeleteFile["k4fixSN2.txt"];
Save["k4fixSN2.txt", fixSN2]
```

--- DeleteFile: Directory or file k4fixSN2.txt not found.