EON GPR Test Systems  
Under the Auspices of REAXPRO

Alejandro, Hannes, Rohit, Satish, Maxim

Brief system descriptions of test cases considered. Also includes developer TODO items.

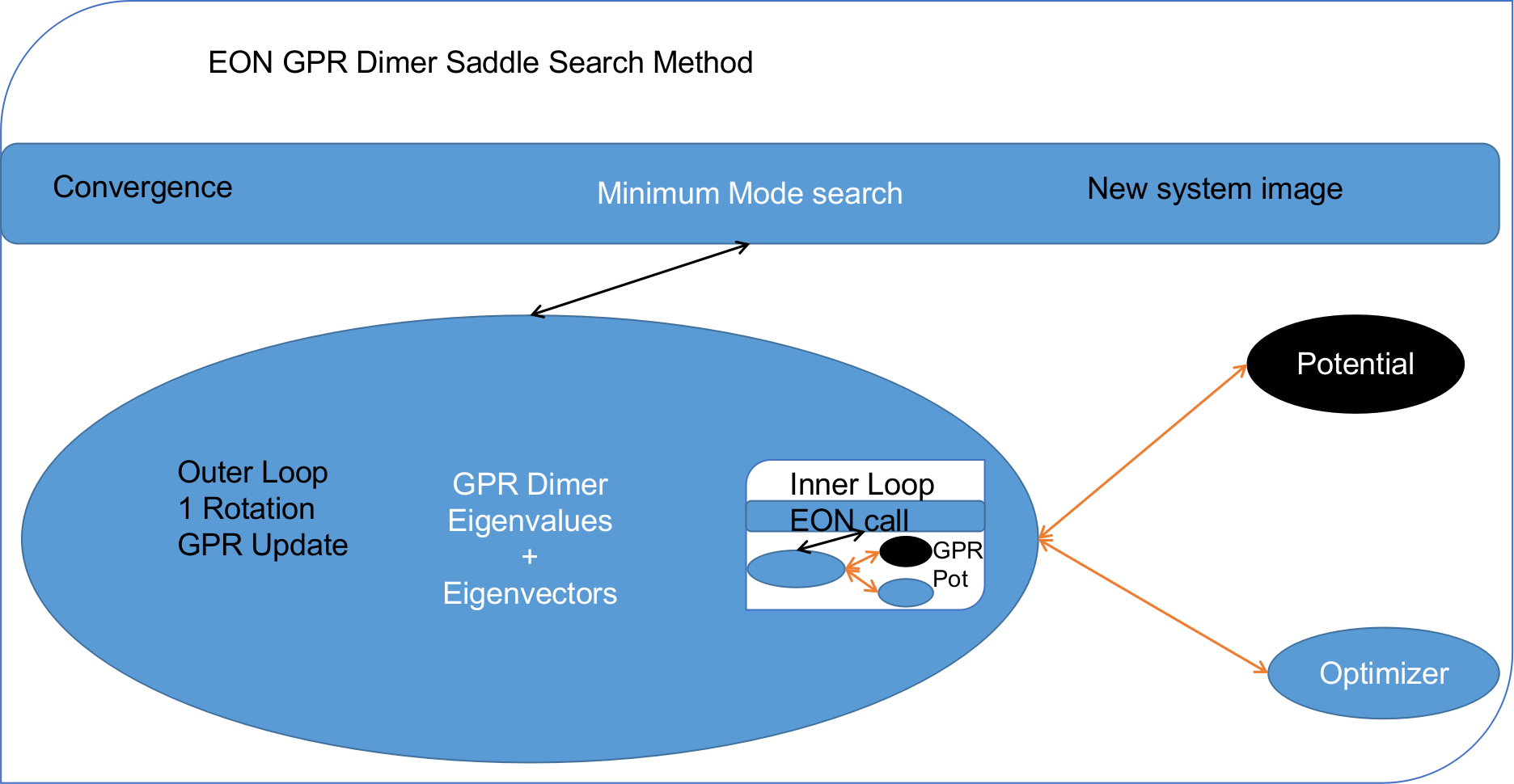
Many of the systems studied here are taken from and the data has been kindly provided by Villi. Before describing the data-sets used, as well as the results; we will give a brief overview of the coupling of eon to the GPR Dimer library. The results of established a baseline for studying the effects of accelerating saddle search methods namely:

* Nudged elastic band calculation
* Tailored bespoke covariance functions for atomic data
* Proof of concept studies of accelerating surface walking methods like the dimer using Gaussian Process Regression

However, these proof of concept studies were implemented in MATLAB and were not amenable to large systems due to both theoretical restrictions and computational inefficiency.

# Architecture

The structure of the interface is given below:



EON to GPR Dimer interface

## Programming Methodology

The GPR dimer is implemented as a stand-alone library (Apache2 licensed) with minimally invasive changes to the EON code. This allowed for the use of modern and performance oriented programming practices like:

* Modern build system
  + Allows for cross-platform optimizations
  + Integrates with MKL for improved HPC performance
* Extensive Testing with the googletest framework
  + Both unit tests and integration tests are provided
  + eon does not include a unit testing framework
    - An extensive set of manual test cases are provided to replicate all existing literature results
* Modular design following OOP practices
  + Separation of GPR hyperparameter optimizations
    - Allows for updating both the minimizer, and the possibility of implementing monte-carlo methods for hyperparameter optimization
  + Separation of GPR surface and the LBFGS call
    - This allows for the usage of other minimizers like the Lanczos
  + Separation of covariance functions
    - A standard interface for the implementation of future covariance functions has been provided
* Only user determined parameters are used internally

We use the numerical linear algebra library, Eigen; which supports Intel MKL optimizations. Eigen is also used by EON.

The library itself has a self contained executable for testing which consumes the same input data as the MATLB code. The interface is seamless, with the addition of input parameters done through the standard eon configuration file.

The structure of the library is such that it accepts a general potential function from EON and then uses it to calculate the GPR approximation to the potential energy surface. For the relaxation phase, the trained GPR potential in EON is called.

A user defined set of pruning parameters have been implemented which ensure optimal and bound scaling of the GPR.

### EON Changes

A Cmake build system was implemented for efficient cross-platform builds. We augment the potentials with an interface to AMS. A GPR potential was also written, which is called by the library for the relaxation phase. Additionally, the build system allows for HPC speed-ups which are determined automatically and uses the MKL.

### User Inputs

All the configurations follow the standard ini file specification of EON. As the gprdimer is implemented as a Minimum Mode Search method; as a replacement for the existing lanczos or dimer methods. Consider the standard configuration for a saddle search run:

[Main]  
job = saddle\_search  
temperature = 300  
random\_seed = 706253457  
  
[Potential]  
potential = morse\_pt  
  
[Optimizer]  
converged\_force = 0.001  
max\_iterations = 1000  
  
[Saddle Search]  
displace\_least\_coordinated\_weight = 1.0  
displace\_radius = 3.3  
displace\_magnitude = 0.01  
min\_mode\_method = dimer  
max\_energy = 10.0

* AMS Interface [sec:org0359ca0]
* [AMS]  
  engine = ADF  
  xc = B3LYP
* The current implementation uses an I/O pipe and supports any of the engines exposed by AMS; however efforts to use the standard [AMSpipe protocol](https://www.scm.com/doc/AMS/Pipe_protocol.html) for AMS are ongoing.
* GPR Dimer [sec:org06aa243] The interface through EON augments the existing conf.ini file with the following GPR specific parameters.
* [GPR Dimer]  
  finite\_angle = 0.05  
  converged\_angle = 0.0873  
  relaxation\_converged\_angle = 0.01  
  max\_initial\_rotation\_iterations = 6  
  max\_relaxation\_rotation\_iterations = 10  
  divisor\_t\_dimer = 10  
  max\_outer\_iterations = 300  
  max\_inner\_iterations = 1000  
  max\_midpoint\_displacement = 0.5  
  rotation\_opt\_method = "lbfgs"  
  translation\_opt\_method = "lbfgs"  
  inner\_opt = "dimer"  
  active\_radius = 10.0  
  dimer\_separation = 0.01  
  convex\_region\_step\_size = 0.1  
  max\_step\_size = 0.1  
  force\_threshold = 0.001  
  ratio\_at\_limit = 0.666666666667  
  nogp\_initial\_rotations = true  
  has\_many\_iterations = true  
  hyperparameter\_opt\_method = "scg"  
  gpr\_variance = 1e-7  
  gpr\_noise\_variance = 1e-5  
  prior\_mean = 0.0  
  prior\_variance = 1.0  
  prior\_degrees\_of\_freedom = 20  
  # OPT parameters  
  opt\_max\_iterations = 400  
  opt\_tol\_sol = 0.001  
  opt\_lambda\_limit = 1e16  
  opt\_lambda\_init = 100  
  gpr\_jitter\_variance = 0  
  # OPT parameters  
  opt\_max\_iterations = 400  
  opt\_tol\_sol = 0.001  
  opt\_lambda\_limit = 1e16  
  opt\_lambda\_init = 100  
  gpr\_jitter\_variance = 0
* Additionally, we have implemented debugging levels in the code for ease of visualization; which are able to generate snapshots of the GPR surface after each call to the true energy and forces (by exhaustively calculating energies over a user defined grid of points around moving atoms) for one dimensional systems.

# GPR Model

The basis.

# **TODO** Toy Systems

These include standard benchmarks for the minimum mode saddle search problems in general.

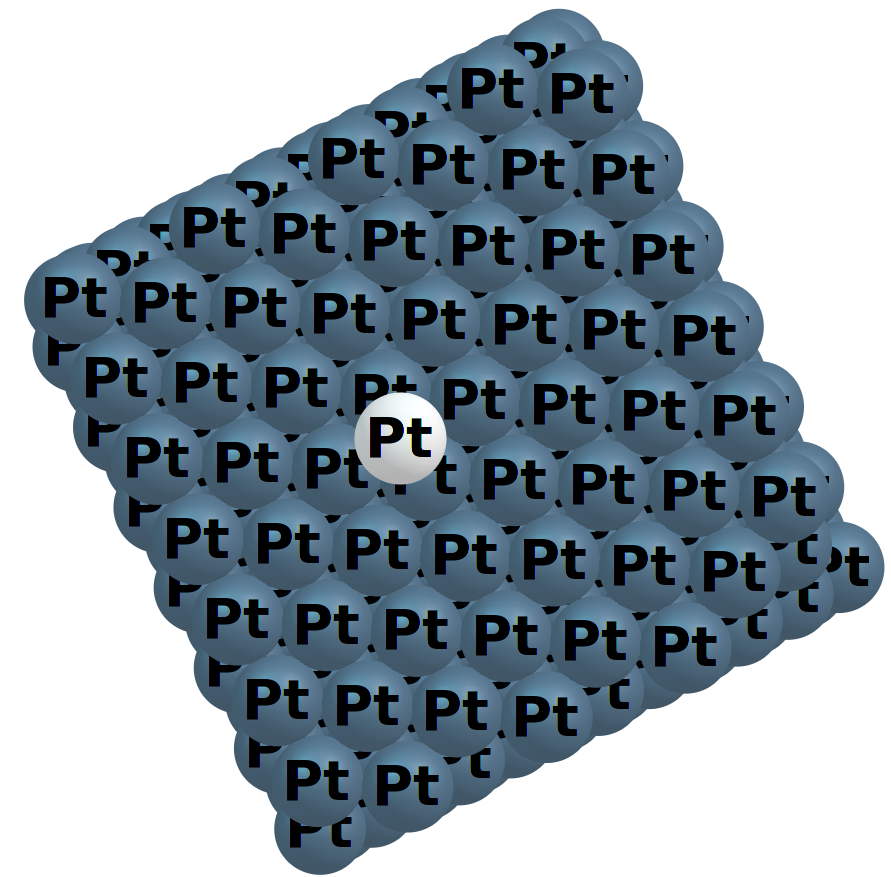
## **TODO** Hydrogen Adatom on Copper

This system was used for porting the code from MATLAB. The C++ code runs 10x faster than the MATLAB code.

## **DONE** Platinum Adatom on Platinum

Chronologically the first system tested with EON. The system has some pleasant qualities which make it easy to work and test:

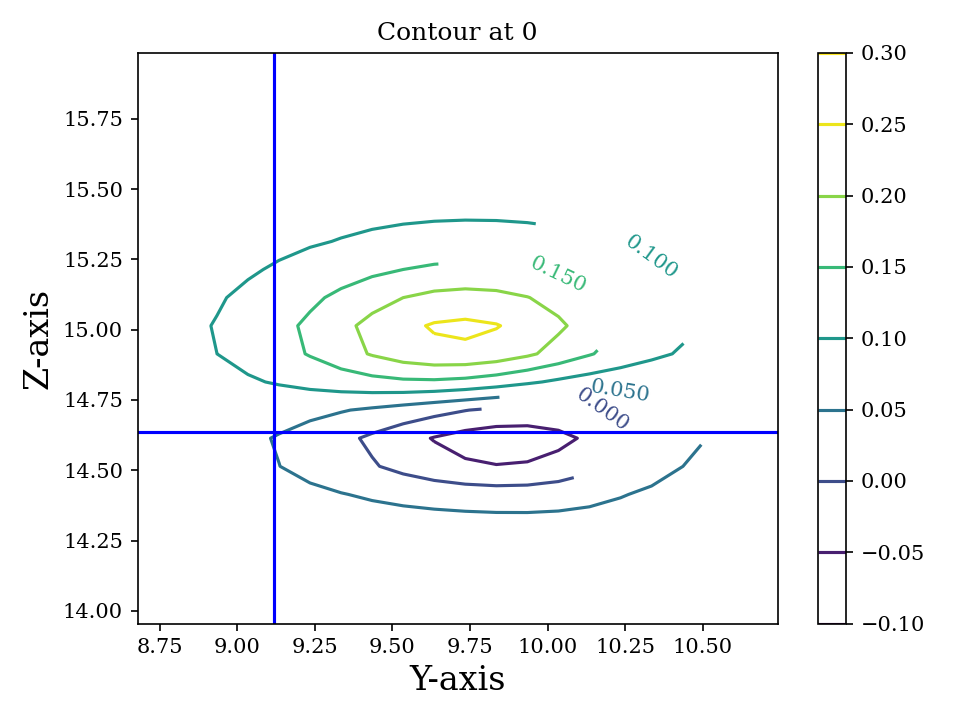
* Easy control of the degrees of freedom
  + Achieved by varying the number of moving atoms and the fixed atoms
* Homogeneity meant neither atomic numbers nor atomic masses were required
* Cheap throwaway potential (Morse)



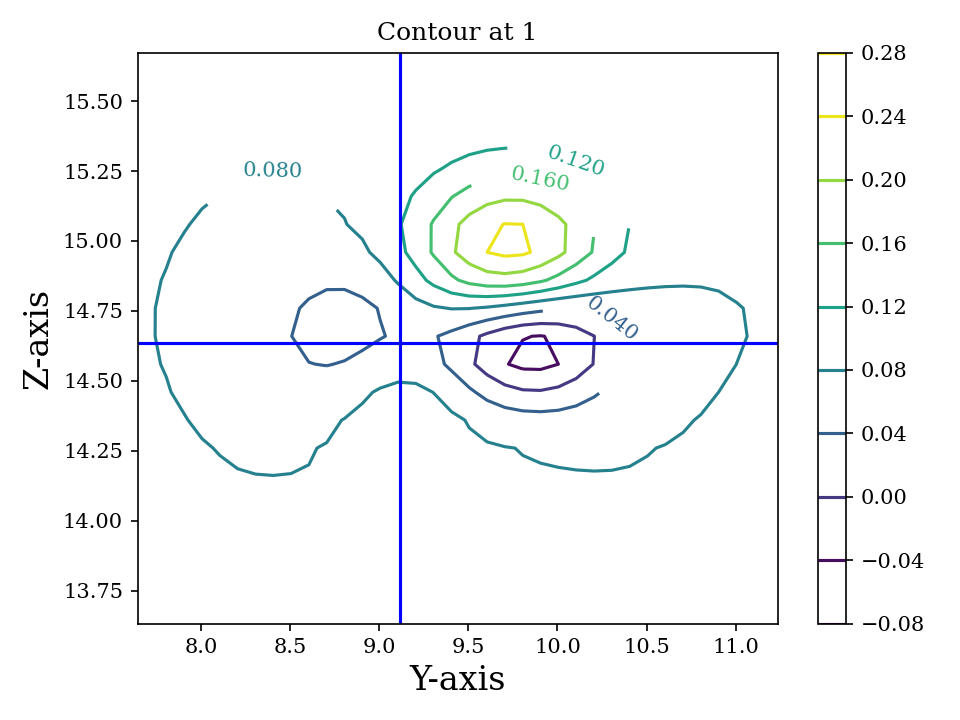
This system takes **0.069s** with the dimer, and **0.109s** with the GPRD in 5 cycles.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Degrees of Freedom | System Size | Active Radius | GPR Time | Dimer Time |
| 48 | 1008 | 5 | 0.109 | 0.069 |
| 12 | 1008 | 3.3 | 0.063 | 0.069 |

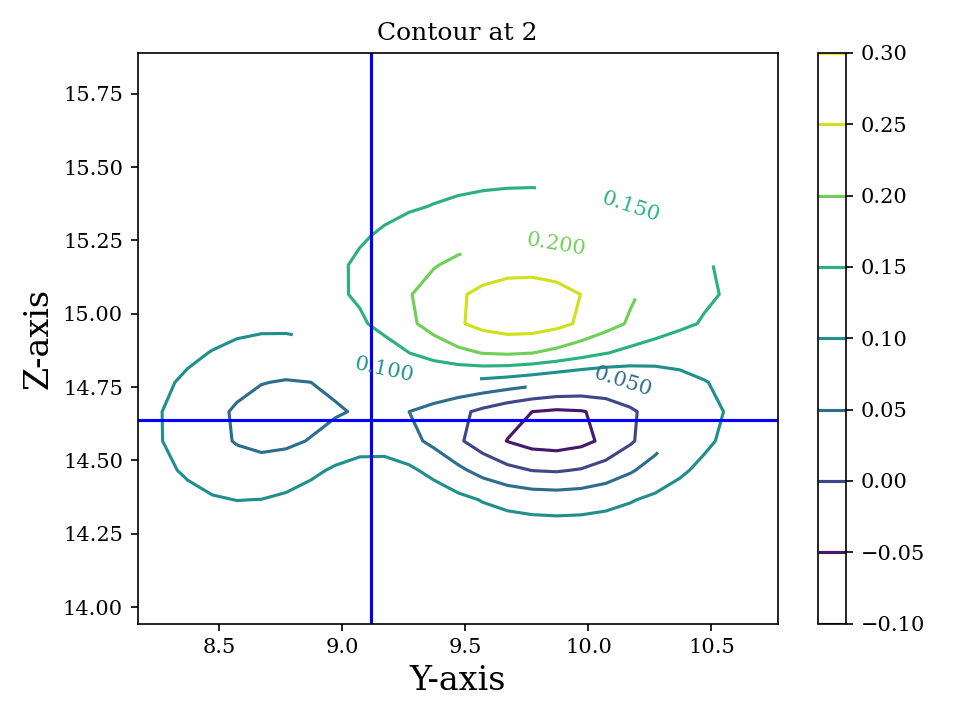
45 degrees of freedom arise from having a cut off radius of 5, which means the surface atoms enter the covariance matrix, and 3 degrees of freedom are from the movement of the adatom itself. For this system the GPR PES has been mapped out as a function of the planar coordinates in Fig. [[optgpr]](#optgpr).



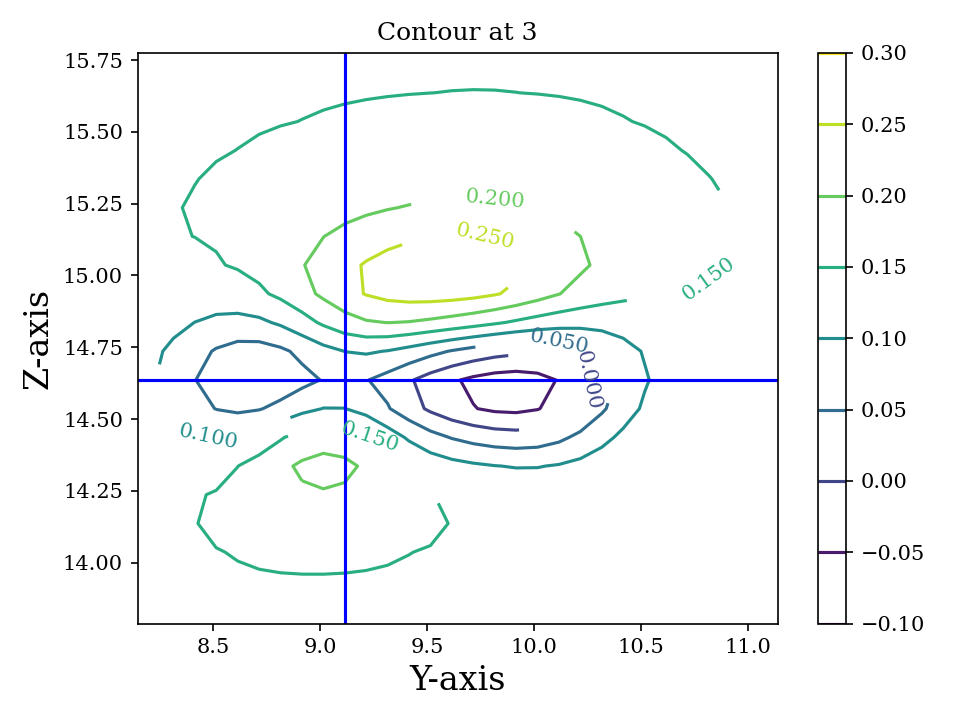
Contour plots representing the GPR PES surfaces predicted for a single Pt adatom on a Pt island with an active radius of 5



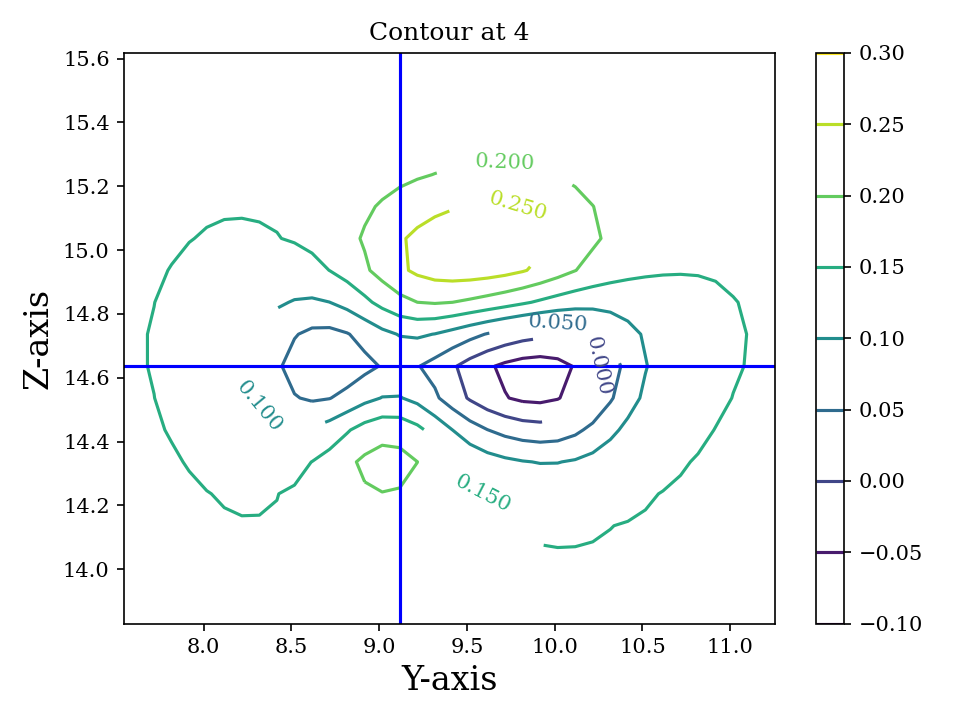
Contour plots representing the GPR PES surfaces predicted for a single Pt adatom on a Pt island with an active radius of 5



Contour plots representing the GPR PES surfaces predicted for a single Pt adatom on a Pt island with an active radius of 5

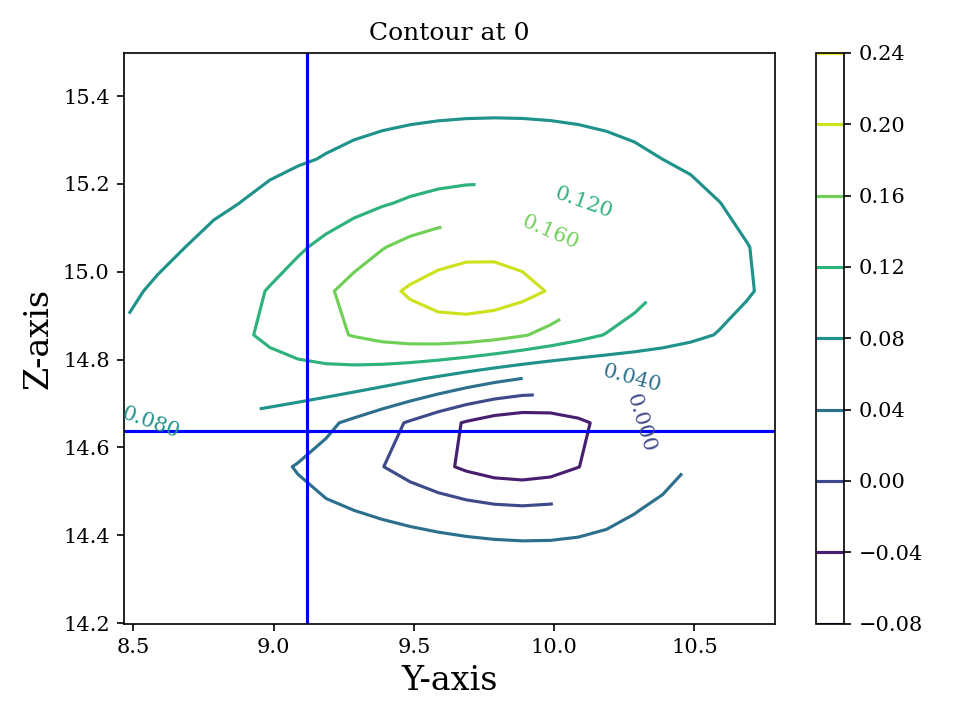


Contour plots representing the GPR PES surfaces predicted for a single Pt adatom on a Pt island with an active radius of 5

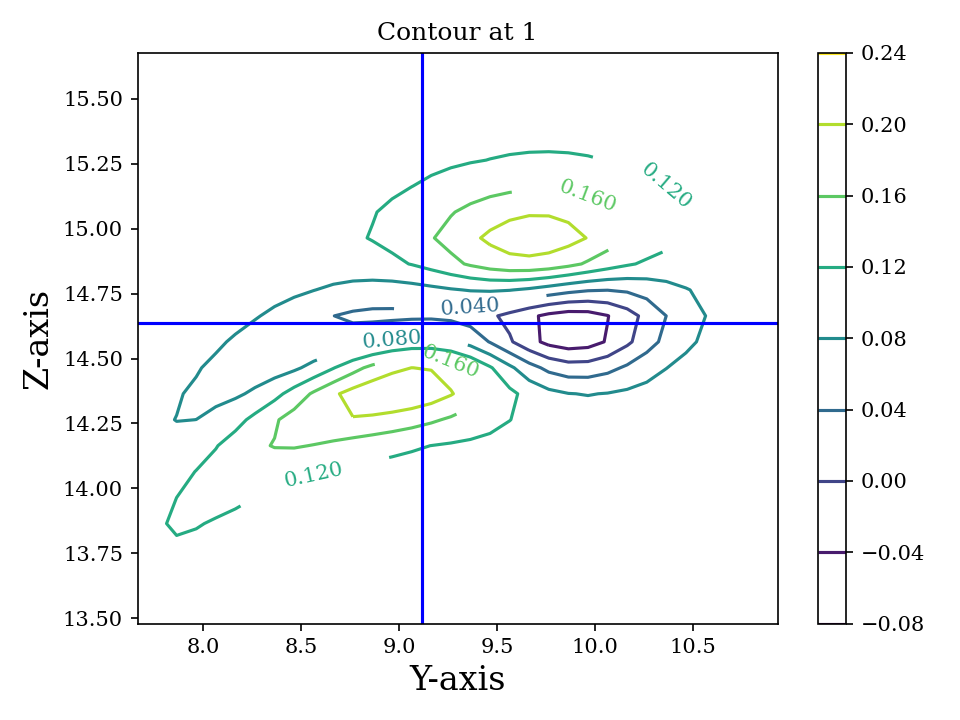


Contour plots representing the GPR PES surfaces predicted for a single Pt adatom on a Pt island with an active radius of 5

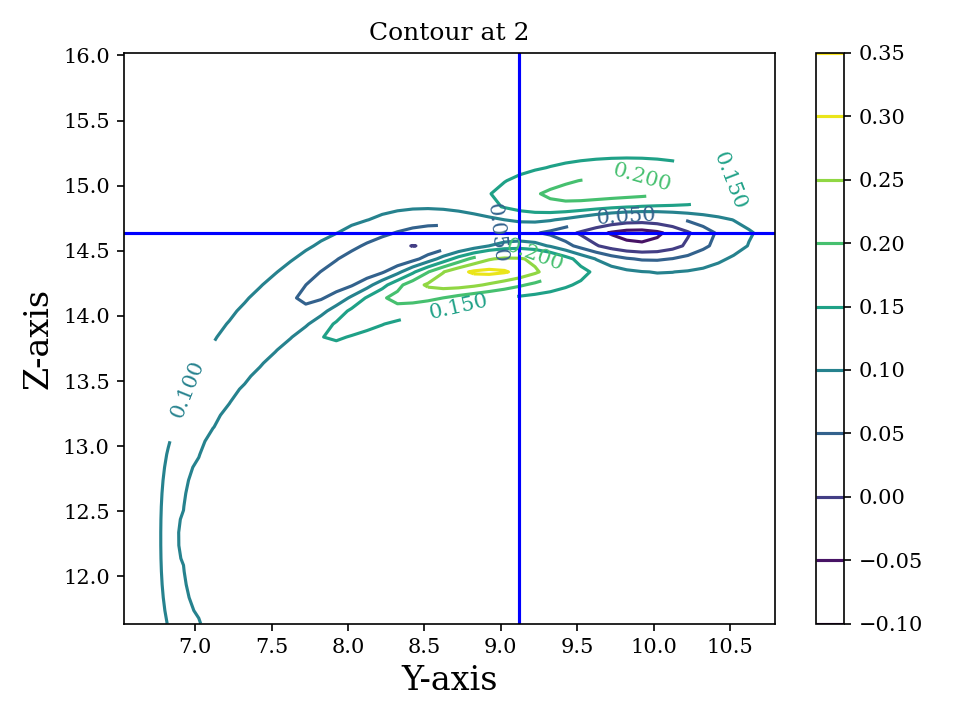
In this instance, the GPRD can converge to the saddle point with a lower active radius of 3.3, however, this takes 9 image evaluations; and the potential energy surface suffers, as can be seen in [[poorsurf]](#poorsurf).



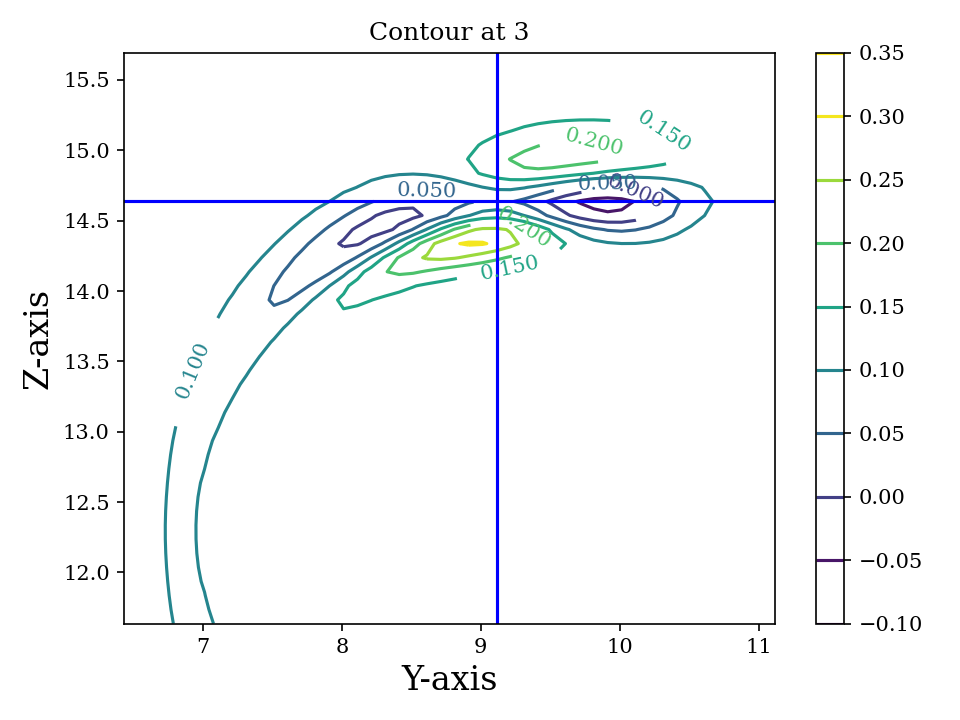
Contour plots representing the GPR PES surfaces predicted for a single Pt adatom on a Pt island with 9 atoms used by the GPR



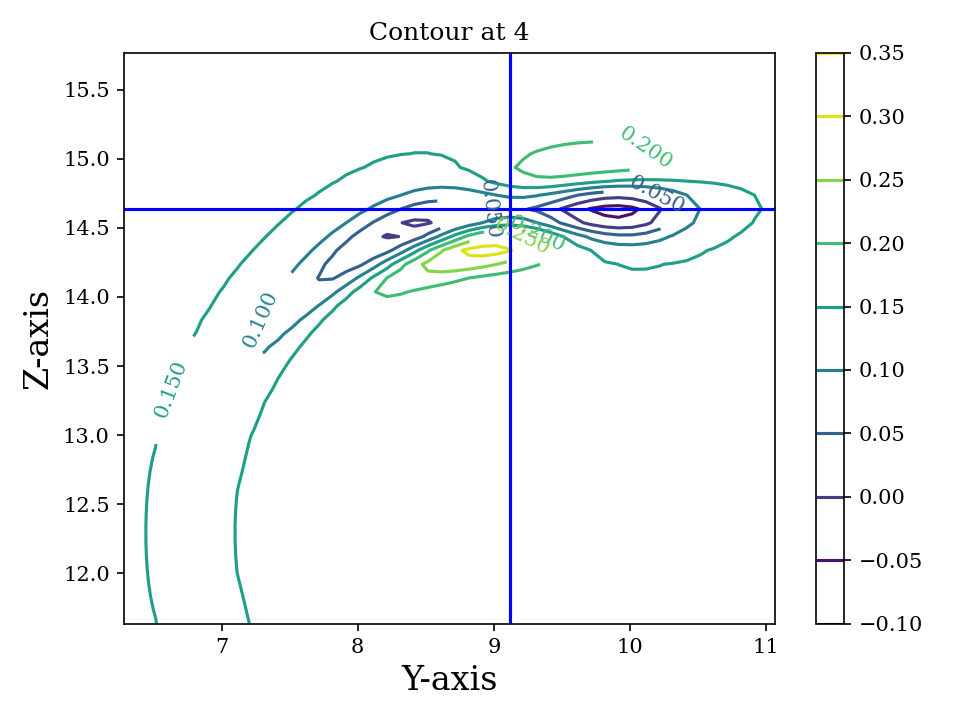
Contour plots representing the GPR PES surfaces predicted for a single Pt adatom on a Pt island with 9 atoms used by the GPR



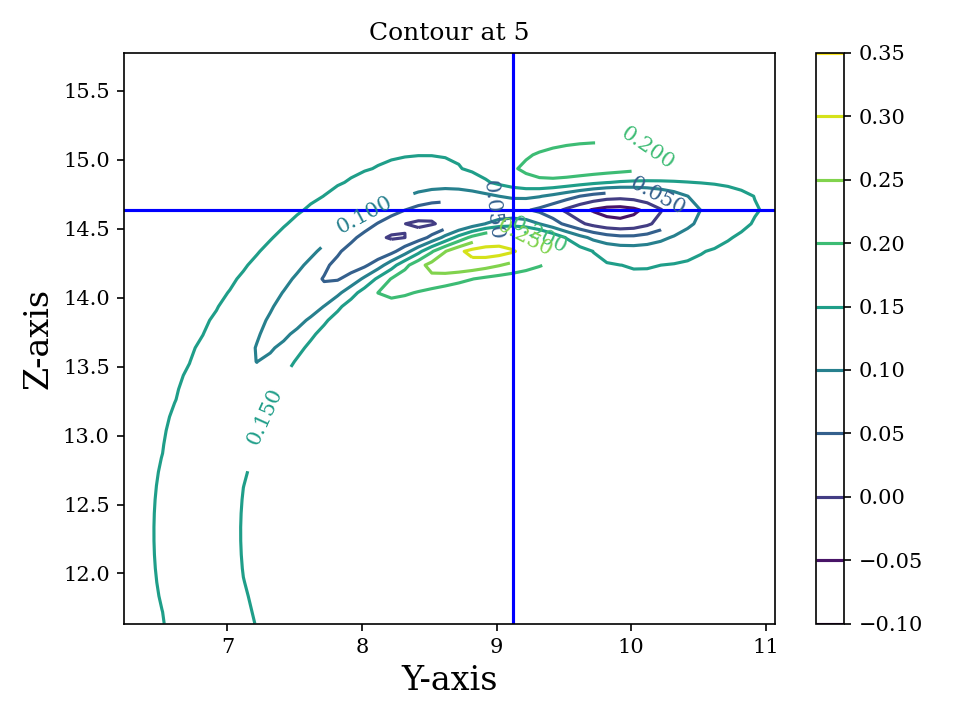
Contour plots representing the GPR PES surfaces predicted for a single Pt adatom on a Pt island with 9 atoms used by the GPR



Contour plots representing the GPR PES surfaces predicted for a single Pt adatom on a Pt island with 9 atoms used by the GPR



Contour plots representing the GPR PES surfaces predicted for a single Pt adatom on a Pt island with 9 atoms used by the GPR



Contour plots representing the GPR PES surfaces predicted for a single Pt adatom on a Pt island with 9 atoms used by the GPR

# **TODO** Literature Extensions

## **DONE** Pruning Rationale

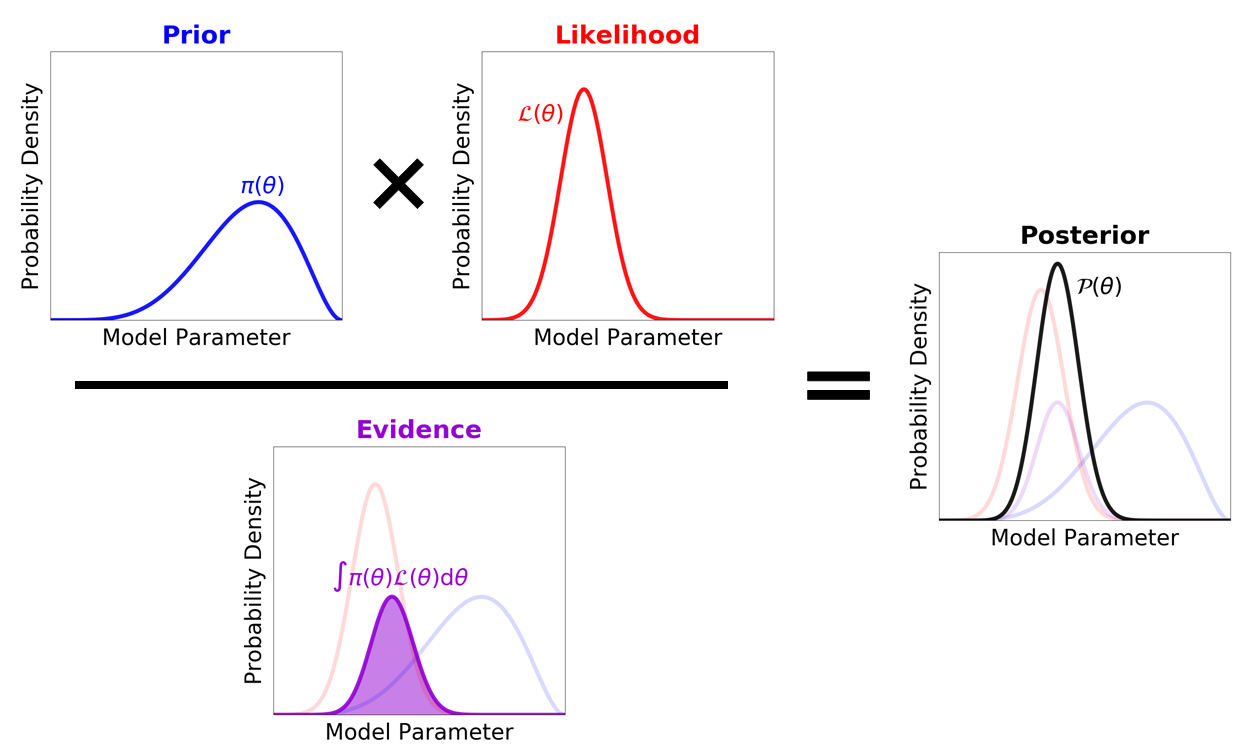
### Hyperparameters and Expectations

For inference, even in higher dimensions; it is not the distribution itself which is important, but the **expectation** value of the the distribution:

$$\begin{aligned}
E\_{\color{blue} p(\theta|y)}[f(\theta)] & = \int f(\theta) {\color{blue} p(\theta|y)} d\theta,\\
\text{where} \quad
{\color{blue} p(\theta|y)} & = \frac{\color{darkgreen}p(y|\theta)p(\theta)}{\color{red} \int p(y|\theta)p(\theta) d\theta}\end{aligned}$$

Where we note that is the posterior probability ; is the likelihood ; is the prior and the normalizing constant is known as the evidence (). The notation of is indicated.

Visually, this master equation (Bayes theorem) can also be visually represented by Fig. [12](#fig:speagleFig).



[fig:speagleFig]Illustration of Bayes theorem

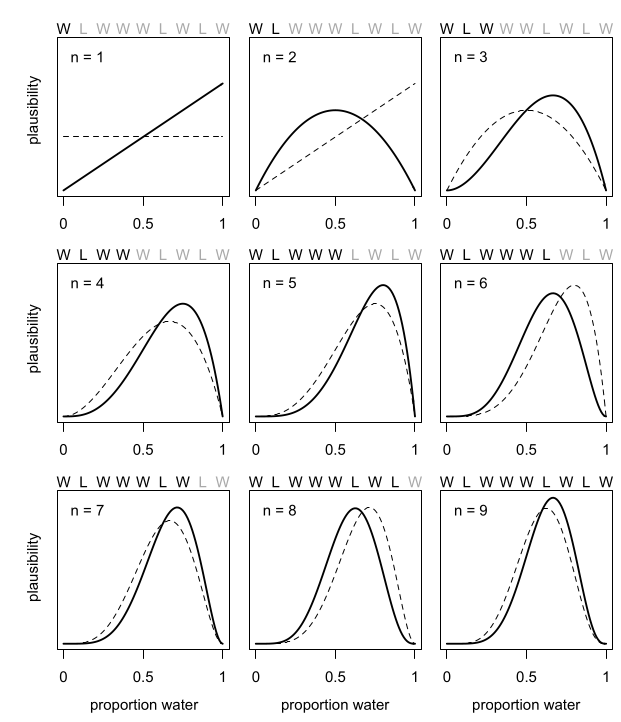
Where we recognize that the marginal probability, or the evidence is the integral over the parameter space; and describes the independent probability of a single event regardless of the others.

None of the preceding discussion implied any limits on the output range of these expectation values, and in a space of functions ; if we consider a regression problem modeled by a Gaussian process we obtain pleasantly simple posterior distribution (a Gaussian) .

Having obtained a GPR model from the observations (which are high dimensional points each consisting of the positions of all points considered, which is of size , which is the potential energy of size and which is of size ), we predict additional points on the GPR surface at a fraction of the cost of calculating the energy and forces from our potential.

### Adaptive Prior Improvements with Pruning

Due to the high dimensional nature of the relevant space of observations; have used a weakly informative (broad) Gaussian prior over the hyperparameters. Empirical studies suggest that the hyperprior distribution; does not have a strong influence on the final predictions; however their studies were in the large data limit and thus are not surprising . Unlike in simulated train-test data sets, as per the algorithm outlined , each call for the true energy and surfaces triggers the formation of new GPR surface (see Fig. [11](#fig:optgpr)). Indeed, given the high cost and low scalability; it is not feasible to reach the high data limit when the effect of the prior is overcome by the likelihood. Most of the models then, are formed in the low data limit, when the priors are of importance. From a frequentist or machine-learning perspective, the prior naturally gives rise to the “regularization” effect , reducing overfitting which is normally explicitly written into models .



[fig:borrowmce]Illustrative effect of iterative modeling from . The lighter dashed line shows the previous model (starting from the uniform prior). Note that importantly, as more data is drawn, the prior becomes less important.

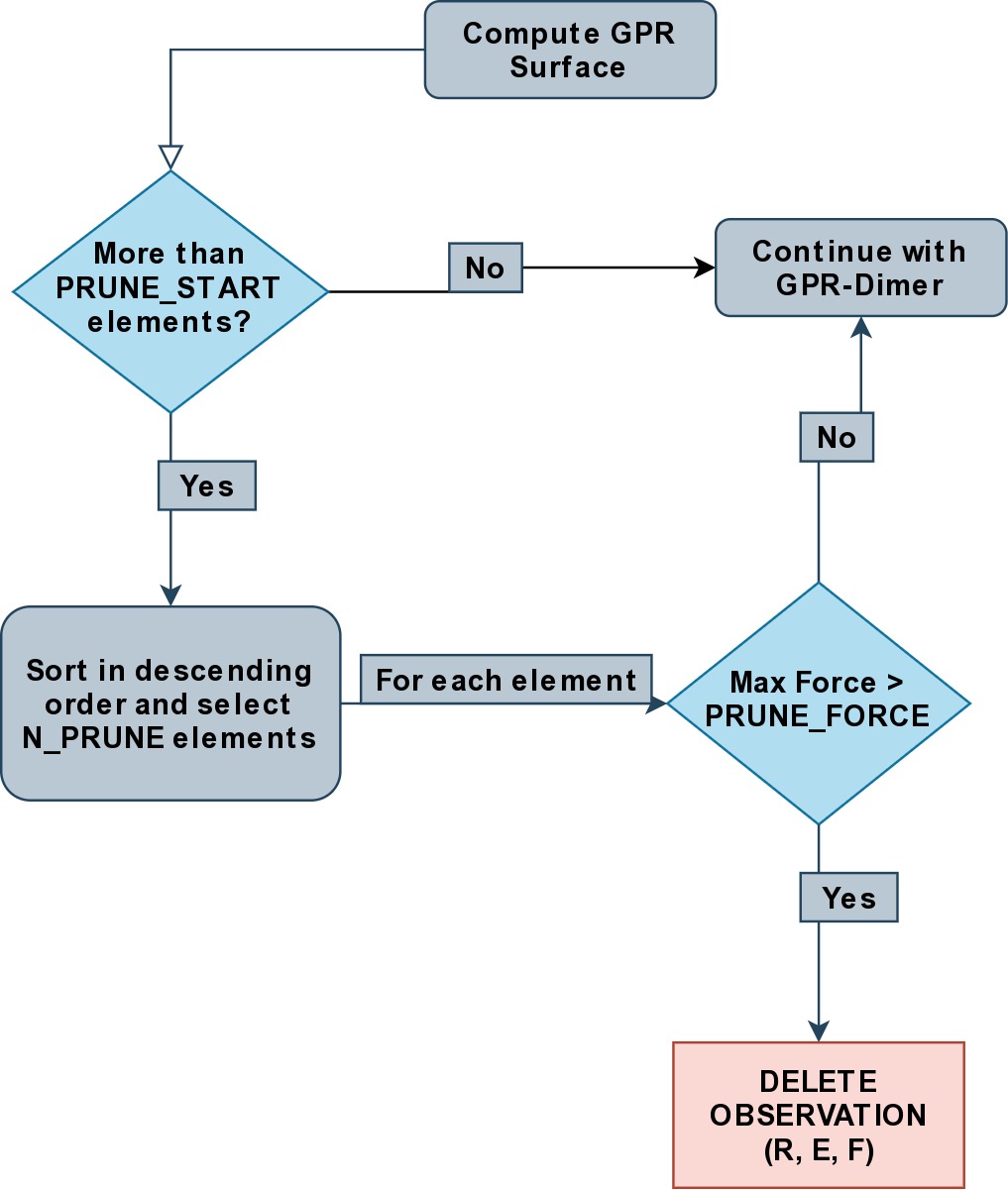
Since we calculate the fitness of each final point on the GPR surface, which is the maximum force in any direction; we can remove poor points from the data. Note that the effect of this in terms of the posterior can be written out for clarity as:

As the product of two Gaussian distributions are also a Gaussian, since we have a Gaussian prior, we can absorb the effect of the second data point into the prior. This means that this iterative modeling could be seen as a case of using a new prior (namely, the posterior of the previous step) with a new data point; which is shown also in Fig. [13](#fig:borrowmce).

Normally, there is no need to compute the model posteriors in this manner, since the model is equivalent, irrespective of being trained one point at a time; or on a set of inputs . However, the effect of poor data is more explicit above; if the likelihood is written out as above. More generally, active data set restrictions have been reported to improve efficiency .

Perhaps more importantly, the local optimization of the hyperparameters scales poorly; with respect to the number of entries in the covariance matrix ; practically, this means that large systems and long timescales are inaccessible in theory as the number of observations increases with each step until convergence. With pruning, we are able to put a bound on the term. Note that overly zealous pruning will lead to too little information being present for the GPR. Adaptive pruning might also help steer the system to saddle points where convergence is difficult. It should be kept in mind that the generation of PES samples (DFT forces and energies) scales worse than typically; and makes the dimer method unfeasible.

By implementing the pruning logic as shown in Fig. [14](#fig:pruneme) using user parameters, we are able to achieve better convergence and in some cases, improved performance as well. Note that since we test for convergence using the maximum forces at the end of each LBFGS translation, we have the results of the fitness function for pruning at a later stage.



[fig:pruneme]Pruning logic embedded in the GPRD to form GPRD-P; NPRUNE is the (max) number of elements to remove, PRUNEFORCE is the force threshold, and PRUNESTART is the starting number of points;

### Quantifying Overheads

We define the GPR overhead to be the difference between the total time elapsed and the number of samples taken from the true PES times the average time for calculating the true forces and energies.

Note that this measure does not account for the time taken by the LBFGS on the GPR surface. We can apply the same formula to the dimer method as well; in order to get an estimate of the normal EON dimer overhead. This means that we can also define a cost function (lower is better) as:

The cost function is per unit time taken for evaluating a single sample of the true potential energy, which means lower is better.

## **DONE** Compute Configuration

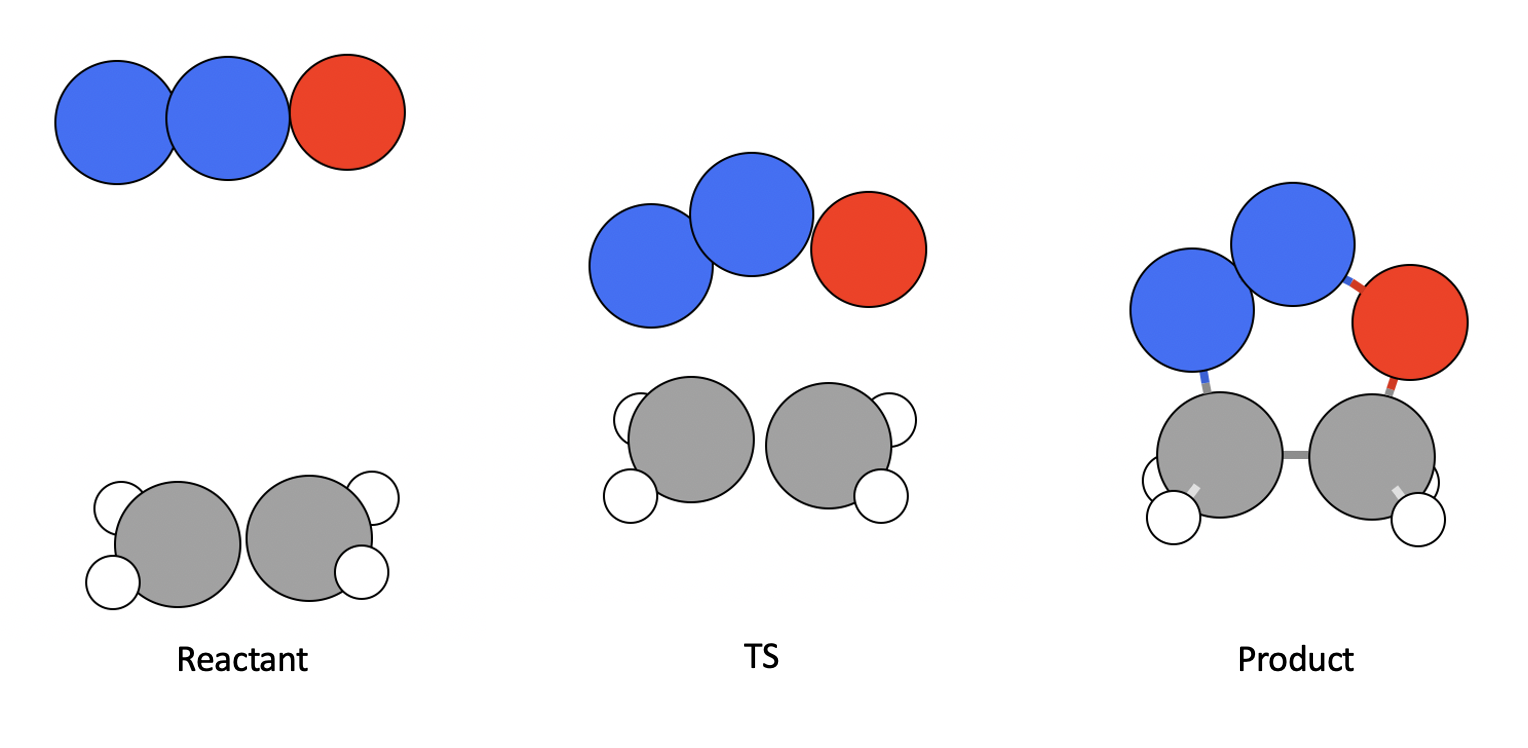
The toy systems above were studied with resources provided by the Icelandic High Performance Computing Centre at the University of Iceland.

All the systems described below were run on individual thin nodes of the Cartesius supercomputing cluster with resources provided by SURFsara. Each node for a run consists of 2 × 16-core 2.6 GHz Intel Xeon E5-2697A v4 (Broadwell) CPUs with 64 GB. All calculations were accelerated with the Intel Math Kernel Library (2020) which provides architecture optimized multi-threaded math functions. The GPR dimer algorithm consists of dense matrix inversions and so distributed memory (MPI) parallelism has not yet been considered for it; however the Amsterdam Modeling Suite was used for the potential energy calculations; which does utilize both the MKL and MPI.

The B3LYP hybrid functional was used which involves a DFT correlation with a combined DFT and Hartree-Fock exchange and is known to reproduce geometries and binding energies of molecular systems with accuracy comparable to Moller-Plesset calculations at lower costs .

## **DONE** Oxadiazole from N2O and Ethylene

Described in .



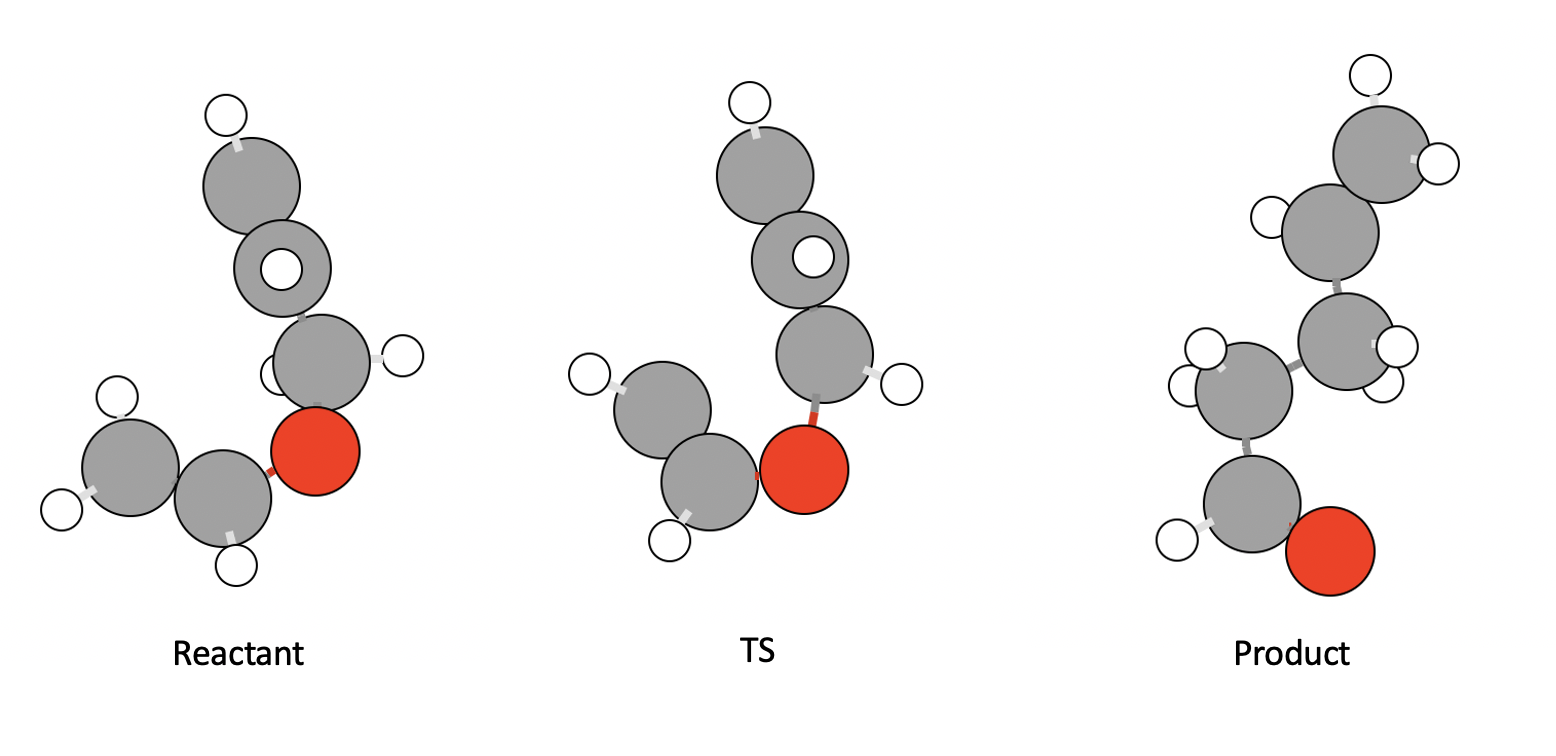
B3LYP-DZ HyOpt is the number of hyperparameter optimizations and HyT is the time taken for hyperparameter optimizations. T indicates time. For the GPRDP the brackets indicated wastage.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Algorithm | Overhead | HyOpt (HyT) | Total T | PES avgT | PES TotT | PES Calls | Cost |
| Dimer | 6512.39 | N/A | 16449.93 | 44.966 | 9937.54 | 221 | 144.554 |
| GPR-Dimer | 242.266 | 15 (53.867) | 1055.516 | 45.181 | 813.25 | 18 | 5.456 |
| GPRD-P | 209.041 | 16 (44.875) | 1065.221 | 45.062 | 856.18 | 19 (8) | 4.672 |

Clearly, the dimer algorithm is sub-optimal for larger systems, while both the GPRD and the GPRD-P perform well. The GPR-P parameters can be tweaked from the defaults to give better performance. The difference in the GPRD and GPRD-P reflects the relative number of LBFGS rotations taken. Note that the scaling term (HyT) is effectively bound by the GPRD-P algorithm, implying that for larger systems the GPRD-P will outperform the GPRD.

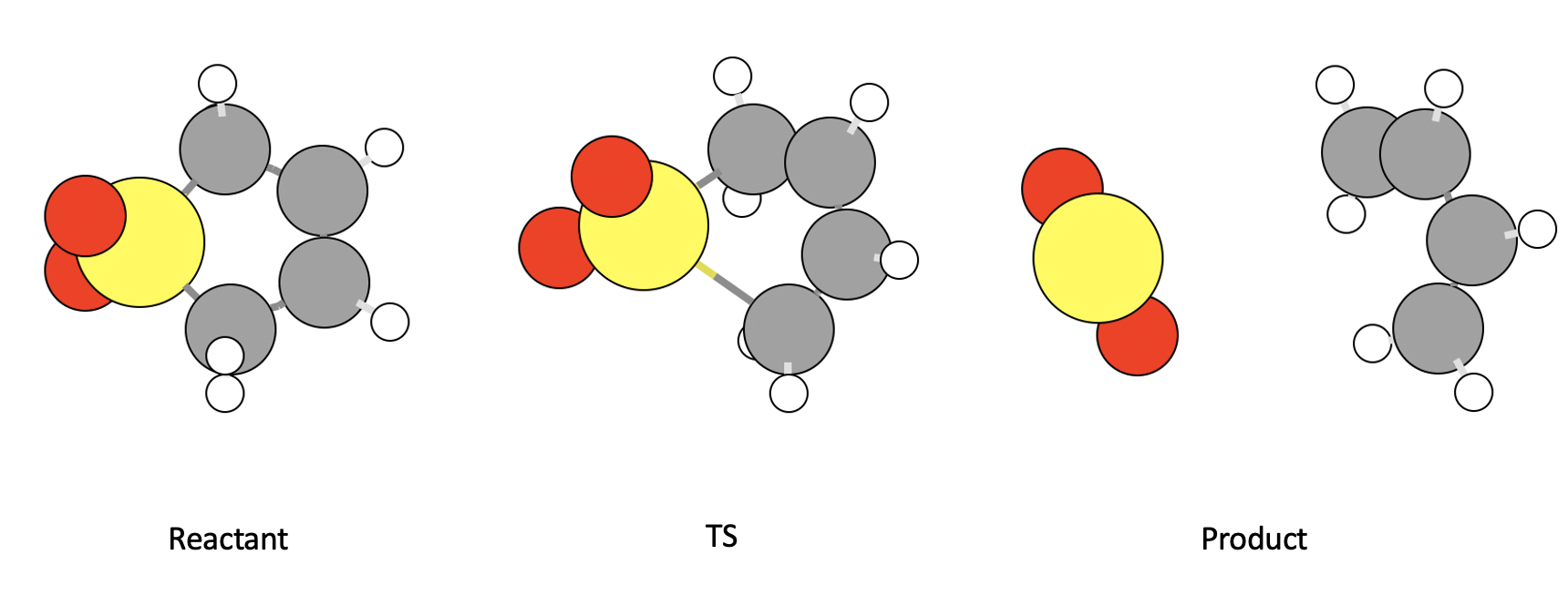
## Allyl Vinly Ether Rearrangement

Also in .



## Sulfolene from Butadiene and sulfur dioxide

Also in .



# Analysis

## Scaling

The training of the hyperparameters using the methods of involves the Cholesky decomposition for the calculation of the inverse of a dense covariance matrix. This operation scales as . However, we show that there are more considerations, including the time taken by the LBFGS on the GPR dimer; which has a fixed scaling in cost per step (like the EON dimer) for a given system, but is unbound in that the time taken to convergence cannot be estimated.

## Pruning

With pruning the covariance matrix size is not allowed to exceed a user-defined size; unless the samples are of a particularly high quality. Pruning is of importance to systems which are larger than those considered here; or for systems which require more iterations.

# References