THERMAL HISTORY INVERSION USING THE FAST GRAIN BOUNDARY MODEL OF DIFFUSION

by

Gabriel Kropf

Submitted in Partial Fulfillment
of the Requirements for the Degree of
Master of Science in Mathematics
with Specialization in Operations Research and Statistics

New Mexico Institute of Mining and Technology
Socorro, New Mexico
May, 2019

This thesis is dedicated to Brian Borchers and Chloe Bonamici, who not only served as research advisors but as motivators and career coaches. Without their help, this work would not have been possible and I would be lost as to where my intellectual passions should be directed. I would also like to thank my committee members, Oleg Makhnin and Rakhim Aitbayev, for their revisions to this document.

And a very special thanks to my parents for their continual support throughout my academic journey.

Gabriel Kropf
New Mexico Institute of Mining and Technology
May, 2019

ABSTRACT

An updated and expanded version of the Fast Grain Boundary (FGB) program, originally developed by Eiler et al. (1994), is presented. In its current form, FGB forward models the oxygen-isotope compositional evolution of a rock resulting from diffusive oxygen isotope exchange. The new FGB program code is written in Python and includes a graphical user interface. Additionally, C compiled versions of the code are available that provide a 20x speedup.

This thesis also describes the inversion of the FGB model to extract thermal histories from oxygen isotope data. The Levenberg-Marquardt (LM) algorithm is applied to the FGB model in search of cooling histories that maximize agreement between the model output and recorded data. Through the use of simulated test cases, it is shown that the LM algorithm is able to distinguish between simple linear cooling and cooling histories containing reheating events. In an effort to maximize the inverse solver's resolution, a heuristic guide to optimal sampling techniques is developed.

This thesis is based upon work supported by the National Science Foundation under Grant No. (EAR-1650355).

CONTENTS

Page

LIST OF TABLES

LIST OF FIGURES

This thesis is accepted on behalf of the faculty of the Institute by the following committee:

Brian Borchers

Academic and Research Advisor

Chloe Bonamici

Research Advisor

Oleg Makhnin

Rakhim Aitbayev

I release this document to the New Mexico Institute of Mining and Technology.

Gabriel Kropf April 15, 2019

CHAPTER 1

INTRODUCTION

Oxygen, being the most abundant element on earth, is a common subject of study among geologists. Understanding its movement and composition within minerals can provide evidence for various earth processes. Oxygen is present in the form of three stable isotopes, i.e. instances of the element with different nucleon numbers, ^{16}O, ^{17}O, and ^{18}O. These isotopes will move within a mineral grain like that shown in Figure 1.1, and between mineral grains adjacent to each other. The isotopic ratios at distinct points within the grain will be continuously affected by this diffusion of isotopes until the mineral is cooled below a cut-off temperature. Once the mineral cools below the cut-off temperature, diffusion has slowed to a point that it is no longer perceptible and the isotopic record becomes 'frozen.'

During the cooling process, the rate of diffusion of isotopes between minerals is dependent on the current temperature. This means that a large amount of information about the minerals' thermal history will be recorded in the isotopic record. The objective of the research presented here is: first, improve the numerical implementation of models simulating the isotope diffusion process; and second, recover the stored information about a mineral's time-temperature history using inverse techniques.

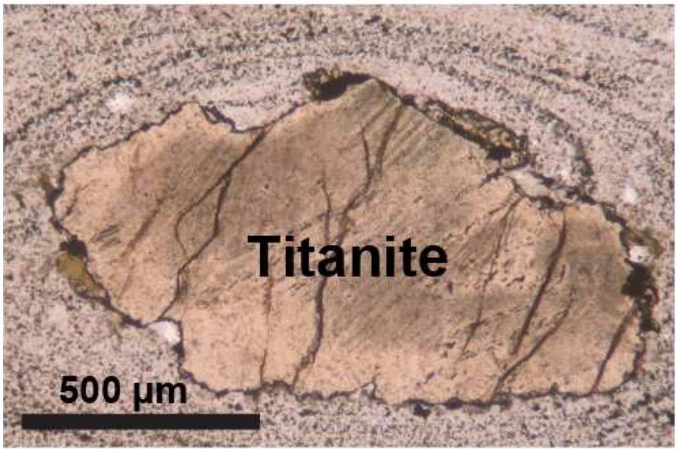


Figure 1.1: Microscopic image of single titanite grain within a mineral sample.

1.1 Background

This background is written to an audience unfamiliar with the methodologies of geochemists. Basic experimental units, measurement techniques, and general problem classification will be given as it relates to the research presented in this thesis.

1.1.1 Isotope Measurements and Standards

Isotopes of elements are denoted by a superscript preceding the element's symbol, where the superscript is the number of nucleons. ^{17}O is an atom of oxygen containing 17 nucleons, which is heavier than one containing only 16, ^{16}O. In general, the composition of isotopes for a given element inside a sample is described by the ratio of the abundance of the heaviest isotope to the lightest. For oxygen, this is the ratio of the number of ^{18}O atoms to ^{16}O atoms within a sample. Isotopic fractionation is the process of partitioning these isotopes between two different substances (or phases). This can result from isotope exchange reactions and kinetic processes ((Hoefs, 2009)). If R_A is the isotopic ratio of a given element for mineral A, and R_B is the same ratio for mineral B, then the partition coefficient is defined as:

$$\alpha_{A-B} = R_A/R_B \tag{1.1}$$

When actually measuring the abundance of isotopes using a mass spectrometer, the precision of *absolute* abundances is significantly poorer than the precision of the *relative* differences in isotope abundances between two samples ((Hoefs, 2009)). For this reason, isotopic compositions are reported using delta-(δ) values, related to the isotopic ratios by Eq. 1.2, where ‰symbolizes parts per thousand.

$$\delta_A = 10^3 \left(\frac{R_A}{R_{\text{standard}}} - 1 \right) ‰ \tag{1.2}$$

Thus, without repeated measurements to reduce statistical variance, we cannot know the isotopic ratio for a given element with greater precision than that of the standards we measure against. For Oxygen, the most widely used standard is Standard Mean Ocean Water (SMOW), which has an uncertainty of 0.46 ‰. The limitations of this precision will be discussed in chapter 4.

Because isotopic ratios vary so little in nature, $R_A/R_s \approx 1$. We can then use the well known approximation $\ln(x) \approx x - 1$ to show that:

$$\frac{1}{10^3}\left(\delta_A - \delta_B\right) = \left(\frac{R_A}{R_s} - 1\right) - \left(\frac{R_A}{R_s} - 1\right) \approx \ln\frac{R_A}{R_S} - \ln\frac{R_B}{R_s} = \ln\frac{R_A}{R_B} \tag{1.3}$$

$$\delta_A - \delta_B \approx 10^3 \ln \alpha_{A-B} \tag{1.4}$$

The equilibrium fractionation factor is then conveniently defined by:

$$\Delta_{A-B} = 10^3 \ln \alpha_{A-B} \approx \delta_A - \delta_B \tag{1.5}$$

Now suppose we have a reference mineral M, and we know the fractionation factors (Δ) between this mineral and two other minerals, A and B. Then we can write:

$$\delta_A - \delta_B = (\delta_A - \delta_M) - (\delta_B - \delta_M) = \Delta_{A-M} - \Delta_{B-M} \tag{1.6}$$

Thus, if we know the individual fractionation factors between a given set of minerals and a common reference mineral, then we also know the difference between any two delta values. Generally, fractionation factors are reported as functions of temperature (T) using the approximation 1.7.

$$\Delta_{X-M} = 1000 \ln \alpha_{X-M} \approx A + (1000)\frac{B}{T} + (1000)^2 \frac{C}{T^2} \tag{1.7}$$

Table 1.1: Summary of common fractionation terms.

Name	Example	
Isotope ratio	R_A	The absolute ratio of the heaviest to lightest isotope in mineral A of an arbitrary element.
Delta value	δ^{18}O	the per mil value of Oxygen-18 in a mineral with respect to a given standard.
Partition coefficient	α_{A-M}	The ratio of isotope ratios between mineral A and monitor mineral M.
Equilibrium fractionation factor	Δ_{X-M}	The difference $\delta_X - \delta_M$. Generally reported as three coefficients A, B, and C.

1.1.2 In-Situ Mass Spectrometry

It is desirable that the element delta values defined in the previous section be recorded at many locations along our mineral grain, as opposed to a single bulk isotopic measurement. This provides significantly more data, and highlights the unique isotopic composition created by distinct cooling histories. In-situ mass spectrometry accomplishes this goal. A traverse across the mineral grain is taken and results in data like that shown in Figure 1.2. The individual ablation pits must be large enough that a sufficient amount of oxygen is liberated for the mass spectrometer to produce accurate measurements of isotope ratios. This limits how close successive measurements can be taken across the traverse, ultimately limiting the resolution of our data.

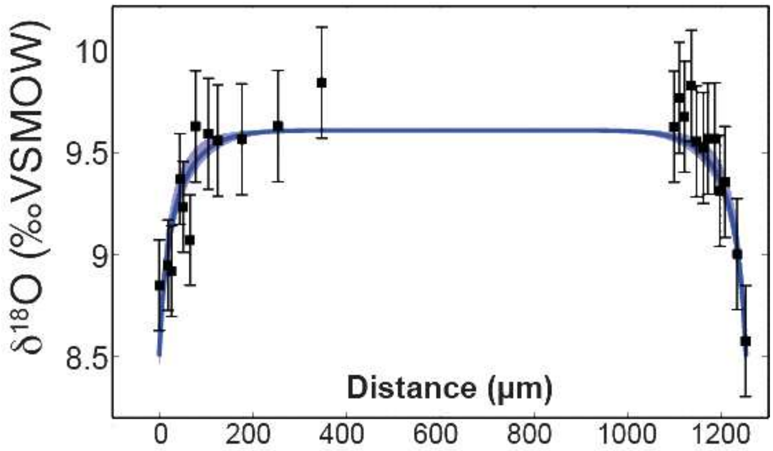


Figure 1.2: Example of isotope data recorded using in-situ mass spectrometry. The black error bars represent the recorded data points along with their analytical uncertainty (2SD), and the blue-line fit was created by visual fit of forward model.

1.1.3 Forward vs. Inverse Problems

In the context of geology, a forward problem is any attempt to replicate a natural process through numerical simulation. This is generally achieved by formulating a concise definition of the process in terms of equations that represent the underlying physics. The equations are then approximated by discretized variables, some number of model input parameters are accepted, and a deterministic set of calculations is carried out that results in outputted data. A prototypical example is the simulation of seismic waves created by an earthquake occurring at any given location. The challenge associated with forward problems is demonstrating that your model appropriately captures the physics. In the case of seismic waves, given the location and intensity of an earthquake, the simulated waves must match with those recorded by any seismometers.

The inverse problem, in contrast, starts with a model and the final data, and attempts to recover the model input parameters. This can prove quite difficult. Just as there are many combinations of speed, angle, and rotation that a pitcher may use to hit the same target, there are potentially many combinations of model parameters that result in the same recorded data. In the case of an earthquake, the inverse problem would be identifying its size and location given a seismograph's readings. Additionally, most inverse methods are iterative and small changes in starting solutions can result in dramatic changes to the final solution.

Thus, for any results from an inversion scheme to be taken seriously, two separate tasks need to be performed. First, the forward model must have been shown to capture the physics, and its results validated through repetitive application to real world data. Second, the inversion scheme should show that the model parameters, or certain features about them, are uniquely determined by the data. And not excessively sensitive to noise in the data.

1.2 Literature Review

This thesis is by no means the first attempt at time-temperature recovery on geological time-scales. The historic achievements in thermal inversions by past researchers will be outlined below. It should be noted that there is a general pattern of initial empirical descriptions for data, followed by forward model development, forward model validation, and finally forward model inversion. However, there does not currently exist any research on the inversion of the Fast Grain Boundary model which was originally developed nearly 25 years ago.

1.2.1 Closure Temperatures

Radiometric dating uses the known rate of decay from a parent isotope into a daughter isotope, along with the two isotopes respective abundances within a system to determine that system's age. However, this analysis works on the assumption that the system under study is closed. Therefore the system's 'time since closure' will be recorded as opposed to its actual age (Jager and Niggli (1964)). For thermally activated diffusion within a mineral, the temperature at which the daughter isotope becomes immobile, and the system therefore becomes closed, is called the closure temperature. It was shown that the closure temperature could be calculated from the parent isotope's decay rate, the diffusivity of the daughter isotope within the mineral, and various geometric parameters (Dodson (1973)). The calculated closure temperature and the radiometric age, together, provide a single point in the mineral's time-temperature history.

The two obvious weakness of this approach to reconstructing a mineral's complete time-temperature history are: first, that a wide variety of minerals must be used to cover the range of potential temperatures encountered. And second, since closure temperatures only provide the *last time* the mineral cooled to the specified temperature, any recovered histories will always be monotonically cooling and reheating events will not be recoverable. To overcome the need for multiple samples "closure functions" for simple geometries were developed that described the closure temperature as a function of position within a cooling object (Dodson (1986)). This allowed multiple points in a system's time-temperature history to be recovered from a single sample. However, still present in Dodson's method of closure temperatures is the assumption that the mineral is exchanging with an infinite reservoir; this is not a physically realistic representation.

1.2.2 Fission Track Lengths and Helium Production

Spontaneous fission within uranium-bearing minerals will leave damage trails referred to as fission-tracks. As the mineral is heated the mineral's crystalline structure will anneal and the fission-tracks shorten. Using previous empirical models for fission-track length reduction and the equivalent-time hypothesis, a complete forward model was developed to describe the forward evolution of fission track length distributions as a function of time and temperature (Green et al. (1989)). Once a mineral is heated above its total annealing temperature though, all fission-tracks will be completely annealed and any thermal information stored will be lost (Ketcham (2005)).

Helium (He) is produced by the decay of both uranium (U) and thorium (Th). It was first proposed that (U-Th)/He data could be used as a thermochronemeter with a range similar to fission-track dating if He loss within apatite was correctly accounted for through modeling low-temperature diffusion (Zeitler et al. (1987)). Using high-precision experiments, the main parameters required to describe He diffusion within apatite were discovered. (Farley (2000)).

Using the foundations for forward modeling both Helium production and fission-track annealing, (Ehlers et al. (2005)) created the HeFTy program for low-temperature thermal history recovery. While the program is exhaustive and easy to use, the method of inversion employed is simple random guessing across a uniform grid of possible thermal histories defined by the user. Thus, the program cannot recover any thermal histories that were not explicitly looked for by the user. Also, due to the physical limitations of both fission-track annealing and He diffusion, the program cannot recover high-temperature histories.

1.2.3 ^{40}Ar/^{39}Ar Step-Heating

In nature, ^{40}K is unstable and decays into radiogenic ^{40}Ar. In a closed system the amount of ^{40}Ar will build up over time. ^{40}Ar/^{39}Ar step-heating is the process of maintaining neutron-irradiated potassium feldspar samples at successfully higher temperatures for set periods of time while measuring the relative rate of ^{40}Ar release. The purpose for first irradiating the samples is to convert ^{39}K into ^{39}Ar and allow the measurement of ^{40}Ar/^{39}Ar, which will have greater precision and improved robustness to sample inhomogeneity, to serve as a proxy for ^{40}K/^{39}Ar measurements. Plotting the relative rate of release against the inverse of temperature will produce an Arrhenius plot. Under the assumption of diffusion from a single domain this plot is expected to be linear (Lovera et al. (1989)).

The multi-diffusion domain model (MDD) was developed to explain the non-linearity in the observed Arrhenius plots of alkali feldspars (Lovera et al. (1989)). The MDD postulated that minerals could have distributions of different diffusion domain sizes (i.e. multiple differently sized pockets of trapped argon gas) and gave a complete theoretically backed model for producing Arrhenius plots that matched those recorded in nature and inverting the plots for continuous time-temperature histories.

More recently, Gallagher (2011) has created the QtQt software which performs time-temperature history inversion using a combination of fission track, (U-Th)/He, and ^{40}K/^{39}Ar data. This is achieved by creating likelihood functions using each data types respective forward model, and employing Markov Chain Monte Carlo (MCMC) to find those thermal histories with the highest likelihood. An advantage to this method is that MCMC produces very visually appealing and easily interpretable confidence regions based on relative likelihoods. However, computationally expensive forward models would not be usable as Gallagher reports using a 100,000 iterations with the first 50,000 discarded as burn in.

1.2.4 δ^{18}O Diffusion Profiles

The diffusive oxygen isotope exchange between coexisting mineral grains results in profiles like that shown in Figure 1.2. Eiler et al. (1994), developed the Fast Grain Boundary model to simulate this isotope exchange using finite difference approximations to the diffusion equation in combination with mass balance constraints. Since the model's conception there has been no work devoted to inverting the model to recover thermal histories or diffusivity parameters from spatially resolved oxygen isotope data.

1.3 Research Objectives

The time-temperature history of a mineral can yield insights about the geological processes that that mineral underwent. In particular large, temperature spikes, or reheating events, are indicative of magmatic injections. Unfortunately, magmatic intrusion temperatures typically exceed fission track annealing temperatures in common minerals used for fission-track dating. Since (U-Th)/He dating was intended as a thermocronometer for similar temperature ranges as those in fission-track dating, neither will effectively store thermal information about the injection. We hope that by computing the inverse of the FGB model, we will be able to recover large reheating events.

The goal of our research is to solve the inverse problem and package the inverse solver into an easily distributable repository so that this type of analysis can be performed by the geological community at large. We will create a large number of simulated test-cases for the solver that are representative of real-world data. The solver will also be tested on 'ideal' cases where the noise level of our data is reduced to demonstrate the potential of our work as the accuracy of mass spectrometry improves. We hope the ideal case solutions will serve as motivation for the geological community to begin creating standards with improved precision. Finally, we will give a heuristic model on optimal sampling procedures for a researcher interested in obtaining the best possible inverse solutions.

CHAPTER 2

THE FAST GRAIN BOUNDARY MODEL

We wish to model the diffusion of stable isotopes through a collection of coexisting mineral grains within a sample. A typical example of mineral grains is depicted in Fig. 2.1; where there are visible grain boundaries separating each individual grain from the rest. Because the diffusivity along these grain boundaries is several orders of magnitude higher than that within the mineral grains themselves, we assume the grain boundaries are always at equilibrium. (Eiler et al. (1994)). Using this fast grain boundary (FGB) assumption we are not required to make any infinite reservoir assumptions.

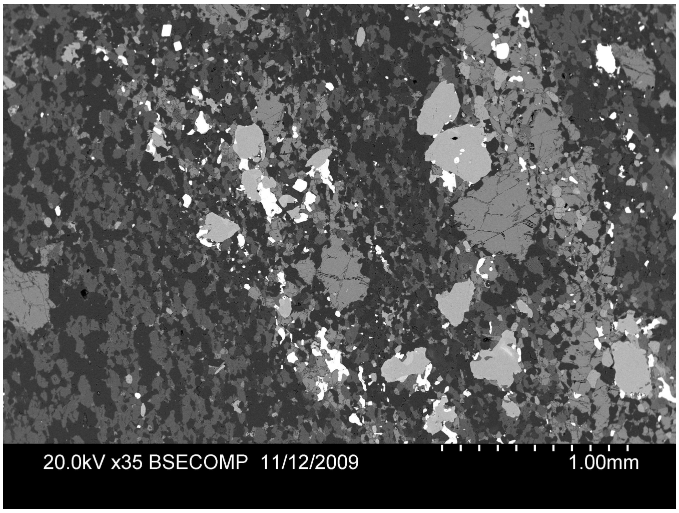


Figure 2.1: Back scatter electron image of a rock sample showing minerals grains. Shades of gray correlate to mean atomic number (density), with lower density minerals appearing darker shades and higher density minerals appearing in lighter shades.

2.1 Background

For modeling the movement of stable isotopes within a mineral, we start with the diffusion equation as given by eq. (2.1). (Shewmon (2016)). This equation describes the most general case in which isotope concentrations are allowed to diffuse within the mineral in all directions; and the mineral may have different diffusivities for each directional flow (anisotropic diffusion). The problem is greatly simplified by reducing our physical model to one dimension, in which we discretize the shape of a mineral by a single parameter. In our work we employ the 1D Cartesian geometry, in which our mineral grain is represented as a slab where oxygen isotopes diffuse along a single axis. We also employ the 1D spherical geometry where the mineral grain is represented as a sphere and oxygen isotopes diffuse radially outward and inward. The spherical geometry simulates the case in which oxygen diffuses at a uniform rate in all directions (isotropic diffusion); and the slab geometry simulates the case in which oxygen diffuses at different rates in one direction vs. another (anisotropic diffusion) (Eiler et al. (1994)). These simplified cases of the general equation are given by eq. 2.2 and 2.3; and the diffusion models they represent are shown in Figure 2.2.

$$\frac{\partial c}{\partial t} = D(t)\nabla^2 c \tag{2.1}$$

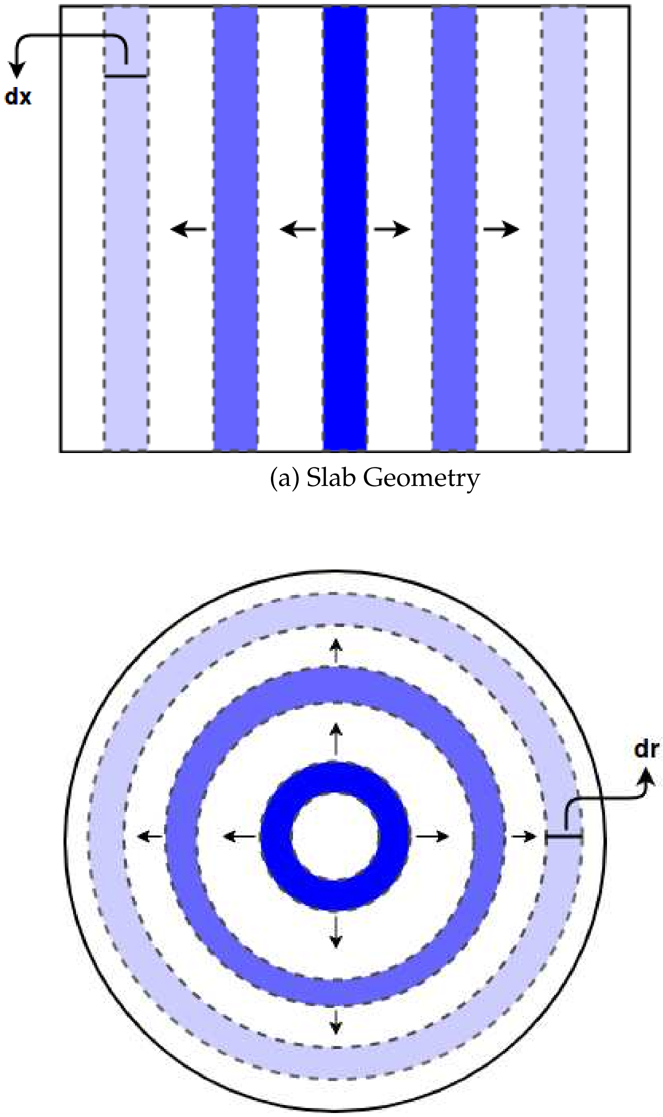
$$\frac{\partial c}{\partial t} = D(t)\frac{\partial^2 c}{\partial x^2} \tag{2.2}$$

$$\frac{\partial c}{\partial t} = D(t)\left(\frac{\partial^2 c}{\partial r^2} + \frac{2}{r}\frac{\partial c}{\partial r}\right) \tag{2.3}$$

If the continuous function u is defined on a finite interval in space, we can discretize it into n nodes representing small intervals of width Δx each. We then solve for an approximate solution by successively solving finite systems of equations. For the linear case, we use the Crank-Nicolson scheme as it has been found to have better accuracy compared to alternative backward/forward-time and central-space schemes (Strikwerda (2004)). This creates an implicit relationship between successive iterations of the function through time. This relationship is shown in Eq 2.4, where $c_{i,j}^{k}$ is the jth node of mineral i at the kth time-step, and $\kappa = D\Delta t/2(\Delta x)^2$.

$$-\kappa c_{i,j+1}^{k+1} + (1+2\kappa)c_{i,j}^{k+1} - \kappa c_{i,j-1}^{k+1} = \kappa c_{i,j+1}^{k} + (1-2\kappa)c_{i,j}^{k} + \kappa c_{i,j-1}^{k} \tag{2.4}$$

The original FGB model developed by Eiler makes use of the fixed-boundary condition, requiring $u_n = u_1 = bc$. Putting this iterative formula, along with the boundary conditions, into matrix form we have the system of equations shown



(a) Slab Geometry

(b) Spherical Geometry

Figure 2.2: Discretized 1D diffusion geometries. Darker shaded areas indicate higher concentrations of ${}^{18}O$ (i.e. higher $\delta^{18}O$ values). Arrows indicate the current direction of diffusion.

in Eq 2.5.

$$\begin{bmatrix} 1 & & & & & \\ -\alpha & 1+2\alpha & -\alpha & & & \\ & -\alpha & 1+2\alpha & -\alpha & & \\ & & \ddots & \ddots & \ddots & \\ & & & -\alpha & 1+2\alpha & -\alpha \\ & & & & & 1 \end{bmatrix}1 \begin{bmatrix} c_{i,1}^{k+1} \\ c_{i,2}^{k+1} \\ c_{i,3}^{k+1} \\ \vdots \\ c_{i,n-1}^{k+1} \\ c_{i,n}^{k+1} \end{bmatrix} = \begin{bmatrix} & \text{grain-boundary} & \\ \alpha c_{i,1}^{k} & +(1-2\alpha)c_{i,2}^{k} & +\alpha c_{i,3}^{k} \\ \alpha c_{i,2}^{k} & +(1-2\alpha)c_{i,3}^{k} & +\alpha c_{i,4}^{k} \\ & \vdots & \\ \alpha c_{i,n-2}^{k} & +(1-2\alpha)c_{i,n-1}^{k} & +\alpha c_{i,n}^{k} \\ & \text{grain-boundary} & \end{bmatrix} \tag{2.5}$$

For the spherical geometry we use the common finite difference discretization with a fixed-boundaries. The matrix form of this discretization is shown in Eq. 2.6.

$$\begin{bmatrix} -2-1/2r & 2 & & & \\ \ddots & \ddots & \ddots & & \\ & (s-1)/s & -2-1/2r & (s+1)/s & \\ & & \ddots & \ddots & \ddots \\ & & & 2 & -2-1/2r \end{bmatrix} \begin{bmatrix} c_{i,1}^{k+1} \\ \vdots \\ c_{i,s}^{k+1} \\ \vdots \\ c_{i,n}^{k+1} \end{bmatrix} = \begin{bmatrix} -1/(2r)c_{i,1}^{k} \\ \vdots \\ -1/(2r)c_{i,s}^{k} \\ \vdots \\ -1/(2r)c_{i,1}^{k} \end{bmatrix} \tag{2.6}$$

One final note on diffusion modeling is that, as a direct consequence of Eq. 2.2, we can calculate the flux into/out of any representative node using the differences of adjacent nodes (Shewmon (2016)). This will be used to ensure mass balance.

2.2 Theoretical Framework for Model

The FGB model is composed of two distinct processes: isotopic diffusion happening within each mineral, and the near-instantaneous isotopic transfer between mineral grains via the boundary pathways. We represent the system by sets of discretized nodes for each individual mineral where $c_{i,j}^{(k)}$ is used to denote the concentration of the jth node of mineral i at time step k. This is depicted in Figure 2.3. The end nodes, representing those parts of the minerals touching the grain boundary, are the only locations where isotopes can flow from one mineral into another. We use the diffusion equation to model isotope movement within a mineral (nodes $i = 2$ through n). And a combination of fractionation factors with mass balance to model the instant equilibration of the grain boundaries (nodes $i = 1$ and $i = n$).

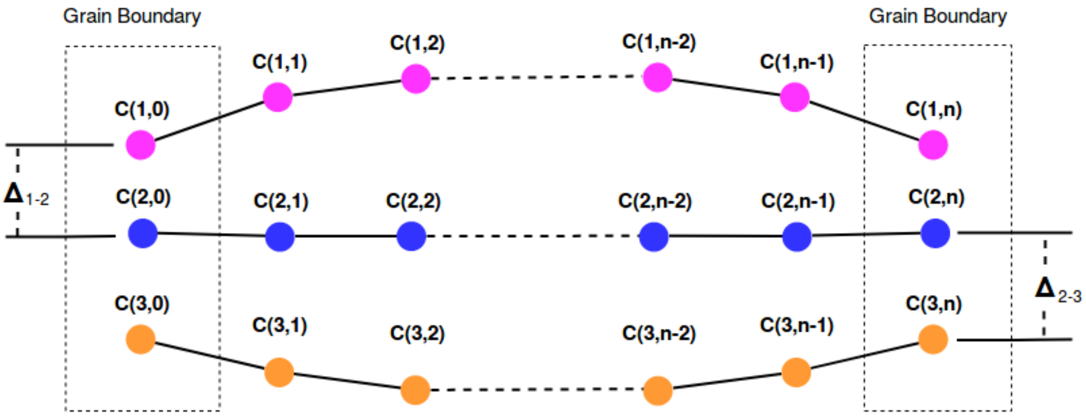


Figure 2.3: Illustration of discretization for diffusion model at a single time-step. C(i,j) is used to denote the concentration of the *j*th node of mineral *i* at the current time-step. $\Delta_{i_1-i_2}$ is the difference between C(i_1,0) and C(i_2,0).

2.3 Numerical Simulation using Python

We have taken the updated MATLAB version of the FGB model created by Chloe Bonamici, and converted it into the open-source language Python. We hope that this, along with an intuitive user interface, will make the inversion software we develop easily used by other researchers. An overview of the numerical forward model will be given here. The next section will describe how all the input parameters are controlled by the user in the actual program, and how simulation plots are created.

The forward model requires as global inputs: the samples whole-rock δ^{18}O value, the simulations time-step and duration, the number of minerals, and a time-temperature history for the sample. Then, we require each mineral's:

(1) diffusive geometry
(2) initial diffusivity
(3) activation energy
(4) mole fraction
(5) fractionation factors (A, B, C), with respect to a common reference mineral
(6) oxygen density
(7) radius
(8) width
(9) spatial resolution (Δx)

Supposing we have n nodes for each of m minerals, at every time-step the simulation will recalculate each mineral's diffusivity and fractionation factors from the current temperature (temperature is provided as a function of time). The representative nodes are then updated in two parts:

i) Diffusive Step (Solving Interior from End Nodes)

For each mineral i, we take a single iteration of the appropriate diffusion scheme for that diffusive geometry individually. Using fixed-boundary conditions, this updates the nodes $j = 2, 3..., n-1$. If mineral i had a linear geometry, this would be accomplished by solving the system of equations 2.7, as outlined in section 1.1.2.

$$\begin{bmatrix} 1+2r & -r & & & \\ & -r & 1+2r & -r & & \\ & & \ddots & \ddots & \ddots & \\ & & & -r & 1+2r & -r \\ & & & & -r & 1+2r \end{bmatrix} \begin{bmatrix} c_{i,1}^{(k+1)} \\ c_{i,2}^{(k+1)} \\ \vdots \\ c_{i,n-1}^{(k+1)} \\ c_{i,n}^{(k+1)} \end{bmatrix}$$
$$= \begin{bmatrix} 2rc_{i,1}^{(k)} + (1-2r)c_{i,2}^{(k)} + rc_{i,3}^{(k)} \\ 1rc_{i,2}^{(k)} + (1-2r)c_{i,3}^{(k)} + rc_{i,4}^{(k)} \\ \vdots \\ 1rc_{i,n-3}^{(k)} + (1-2r)c_{i,n-2}^{(k)} + rc_{i,n-1}^{(k)} \\ 1rc_{i,n-2}^{(k)} + (1-2r)c_{i,n-1}^{(k)} + 2rc_{i,n}^{(k)} \end{bmatrix} \tag{2.7}$$

ii) Mass Balance (Solving End Nodes From Interior)

From the m minerals, we have $m-1$ constraints for the current end node concentrations using the given fractionation factors. Using Fick's second law, we can calculate the flux of δ^{18}O out of each mineral grain by equation 2.8, where D_i is the diffusivity of mineral i. The mass balance equation is given by 2.9, where χ_i is the volume fraction, A_i is the surface area, and ν_i is the oxygen density.

$$J_i^{(k+1)} = -\frac{D_i^{(k+1)}}{\Delta x_i}\left(c_{i,n-1}^{(k)} - c_{i,n}^{(k+1)}\right) \tag{2.8}$$

$$\sum_{i=1}^{m} \chi_i A_i \nu_i J_i^{(k+1)} = 0 \tag{2.9}$$

Then substituting 2.8 into 2.9, results in the final mth constraint that is a function of concentration as shown in equation 2.10. Combining all m equations we get the system (2.11), where $k_i = \frac{D_i}{\Delta x_i}\chi_i A_i \nu_i$. Solving this system represents the equilibration of the boundary pathways and ensures mass balance is maintained through out the diffusion simulation.

$$\sum_{i=1}^{m} \chi_i A_i \nu_i \left(\frac{D_i^{(k+1)}}{\Delta x_i} c_{i,n-1}^{(k)}\right) = \sum_{i=1}^{m} \chi_i A_i \nu_i \left(\frac{D_i^{(k+1)}}{\Delta x_i} c_{i,n}^{(k+1)}\right) \tag{2.10}$$

$$\begin{bmatrix} 1 & -1 & 0 & \dots & 0 \\ 1 & 0 & -1 & \dots & 0 \\ \vdots & & & & \vdots \\ 1 & & & & -1 \\ k_1 & k_2 & \dots & \dots & k_m \end{bmatrix} \begin{bmatrix} c_{1,n}^{(k+1)} \\ c_{2,n}^{(k+1)} \\ \vdots \\ c_{m-1,n}^{(k+1)} \\ c_{m,n}^{(k+1)} \end{bmatrix} = \begin{bmatrix} \Delta_{1-2} \\ \Delta_{1-3} \\ \vdots \\ \Delta_{1-m} \\ \sum_{i=1}^{m} k_j c_{i,n-1}^{(k)} \end{bmatrix} \tag{2.11}$$

Finally, the general algorithm for the FGB model can therefore be written as shown in Algorithm 1 below.

Algorithm 1 Forward FGB Model

```
procedure INITIALIZE
    time = 0
    tend = duration/$\Delta t$
    Solve for initial equilibrium concentrations
for t=1, 2,..., tend do
    time = time + $\Delta t$
    Get current temperature $T$
    for j=1, 2, ..., m do
        Calculate diffusivity $D_j$ from $T$
        Solve system 2.7
    Calculate fractionation factors $\Delta_{1-2}$,...$\Delta_{1-m}$ from $T$
    Solve system 2.11
Return $m$ individual isotope concentration profiles
```

The Python code for the forward model can be found inside *modelfunctions.py* from the Github repository. The code takes a temperature history as the input, and produces as output a diffusive profile like that shown in the blue line of Figure 1.2. The code takes, on average, 4-6 seconds to run a typical simulation.

2.4 Performance Gains using C

Running just the forward model simulations, the Python code used in the previous section works fine with a run-time of roughly 4 seconds. However, when running the inverse solver we need to calculate the Jacobian of our error function. This requires anywhere from 10-30 forward runs per iteration, depending on the resolution of our solution. This results in 20+ minute run-times. If we then wish to use a multi-start approach, this becomes unmanageable. Using a basic profiler we see that the majority of the model's run-time is spent solving the system of equations within the diffusion step of our simulation. Therefore, this is where we focus our effort in optimizing the code.

One of the main issues with Python is that it is an interpretive language. Thus, when attempting to use problem specific algorithms, the interpretive overheard of defining functions outweighs the computational savings. The diffusion schemes we use create tridiagonal systems of equations, for which Thomas algorithm is highly efficient in terms of flops, or computations. We tried SciPy's generic system solver, SciPy's banded matrix solver, and a custom function employing Thomas' algorithm. Figure 2.4 shows the runtime vs. matrix size using randomly generated tridiagonal matrices. Typically, the spatial resolution we use results in matrices of size 200 by 200. We can see from the plot that while the more general banded solver is not as efficient as Thomas Algorithm, it performs better because it is a built-in function that has already been compiled with C. From a more detailed profiler, is was found that the diffusion solver is spending 50% of its time just generating the matrices and vectors used for the actual SciPy function calls.

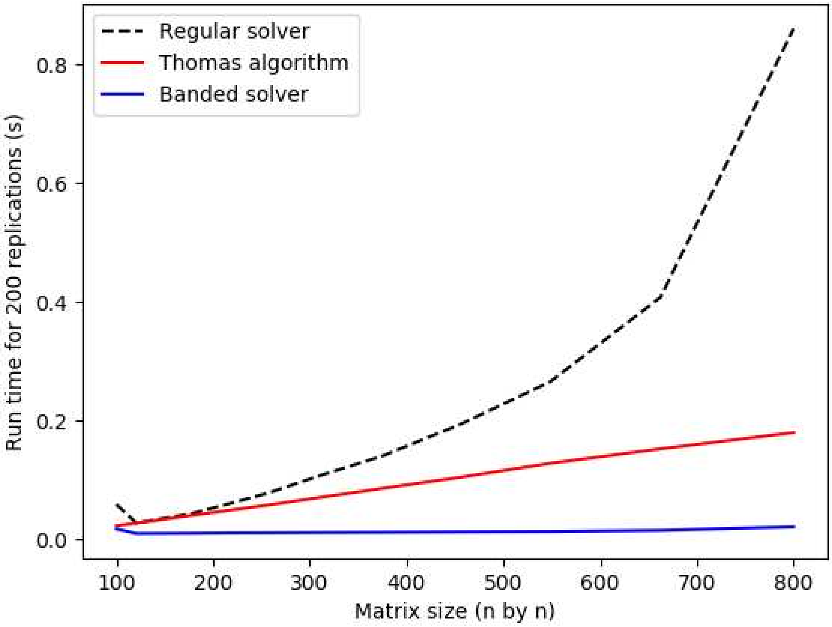


Figure 2.4: Comparison of run times for matrix inversion using different solvers.

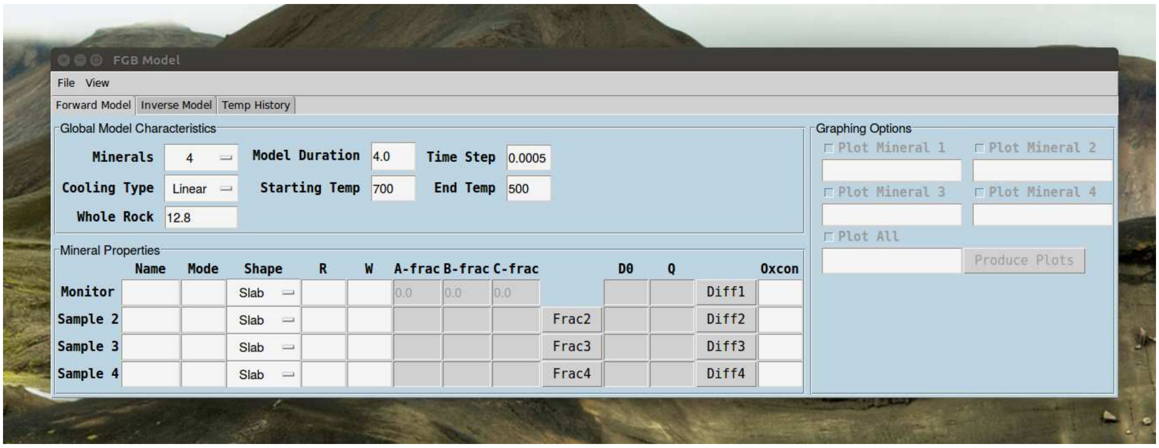
Therefore, building the forward model in C allows us to save significant time in declaring variables, and we are able to employ a more problem-specific algorithm for the solver. The C code is written to accept file locations as input from the command line; the os module in python is then used to execute the compiled

C code. This allows us to avoid defining Cython structs for arrays, and allows the C code to be compiled with the highest level of optimization. Additionally, the repetitive calculations in constructing our tridiagonal systems of equations are stored in C arrays. Then using C's built-in memcpy function we are able to save time by avoiding rebuilding the full array at every iteration. The actual oxygen-isotope profiles at each iteration are stored on the heap, but all reference parameters are kept on the stack to maximize memory transfer speeds. The end result is that the new C code cuts the runtime of forward mode simulations down from 4s to roughly .2s.

2.5 Using the Python GUI - Forward Model

The forward model accepts global model parameters, mineral-specific parameters, and an expected time-temperature history for the mineral sample. The model output will be the isotope composition for each mineral grain in our sample. This section will walk through a basic setup. The files referenced can be download from the repository https://github.com/gkropf/FastGrainBoundary-DiffusionSolver.

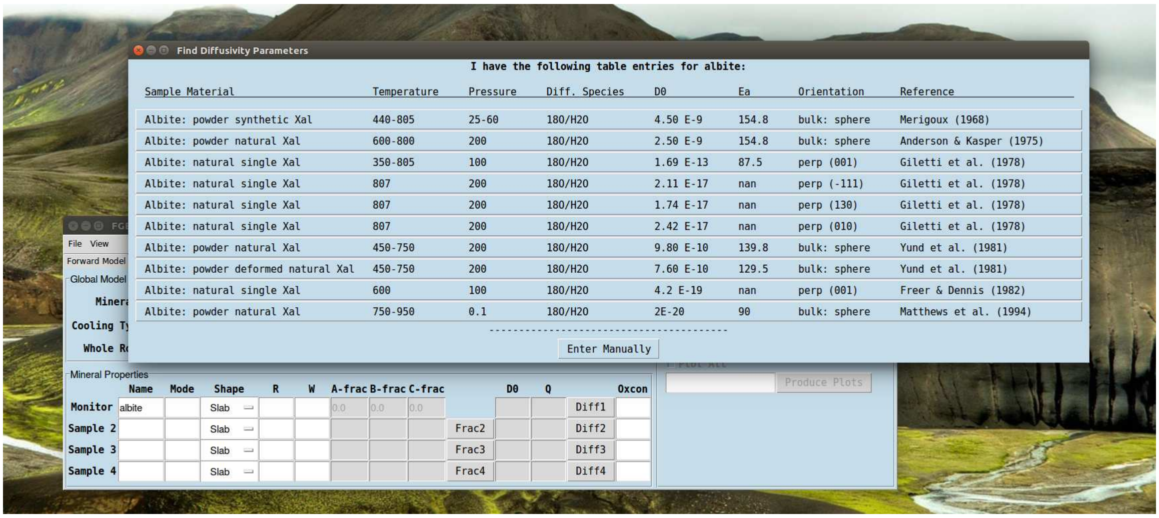
On the first tab of the application entitled ”Forward Model”, we will set the number of minerals to four. Current version allows for any number between two and eight. Next, we will select a linear cooling type; this creates a cooling history where temperature is proportional to time. With this we will need to specify the starting and ending temperatures, and the time duration of this cooling. There is also a reciprocal cooling option (temperature inversely proportional to time), and a custom option. The custom option will ask for a comma-delimited text file containing times in the first column and cooling rates for that corresponding length of time in the second column. All times are in millions of years, and all temperatures are in Celsius unless otherwise stated. We will also need to specify the whole rock delta-18O of our sample. After putting in the general model parameters you should have the following screen:



Minerals 4 | Model Duration 4.0 | Time Step 0.0005
Cooling Type Linear | Starting Temp 700 | End Temp 500
Whole Rock 12.8

Figure 2.5: Global model parameters in forward model.

To input diffusion parameters type in the name of your mineral, and then click the corresponding 'Diff' button. This should cause the menu shown in Figure 2.6 to pop-up. You can select from the any of the current options or enter manually. Similarly for fractionation factors, enter the name of your mineral and the monitor mineral and the program will look for any chain of references that will relate the two. If there are multiple options for any step then the program will allow you to select which one to use. After putting in the modes (mole fractions), mineral radii and widths, and the shape parameters describing the diffusion geometry, we have the window shown in Figure 2.8. You can use the 'File−>Load Parameters' option to import these parameters from the *Examples/Ex_Params1.txt* file.



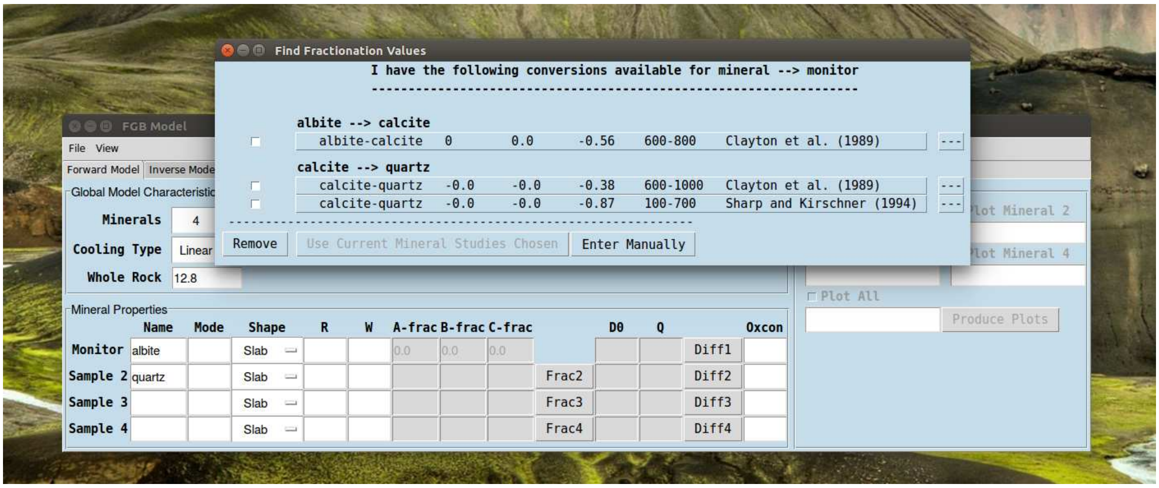
I have the following table entries for albite:

Sample Material	Temperature	Pressure	Diff. Species	D0	Ea	Orientation	Reference
Albite: powder synthetic Xal	440-805	25-60	180/H20	4.50 E-9	154.8	bulk: sphere	Merigoux (1968)
Albite: powder natural Xal	600-800	200	180/H20	2.50 E-9	154.8	bulk: sphere	Anderson & Kasper (1975)
Albite: natural single Xal	350-805	100	180/H20	1.69 E-13	87.5	perp (001)	Giletti et al. (1978)
Albite: natural single Xal	807	200	180/H20	2.11 E-17	nan	perp (-111)	Giletti et al. (1978)
Albite: natural single Xal	807	200	180/H20	1.74 E-17	nan	perp (130)	Giletti et al. (1978)
Albite: natural single Xal	807	200	180/H20	2.42 E-17	nan	perp (010)	Giletti et al. (1978)
Albite: powder natural Xal	450-750	200	180/H20	9.80 E-10	139.8	bulk: sphere	Yund et al. (1981)
Albite: powder deformed natural Xal	450-750	200	180/H20	7.60 E-10	129.5	bulk: sphere	Yund et al. (1981)
Albite: natural single Xal	600	100	180/H20	4.2 E-19	nan	perp (001)	Freer & Dennis (1982)
Albite: powder natural Xal	750-950	0.1	180/H20	2E-20	90	bulk: sphere	Matthews et al. (1994)

Monitor albite

Figure 2.6: Search menu for diffusion parameters.

To save the model parameters use the 'File−>Save Parameters' button. These saved model files will be used for the inverse solver explained below. Now click



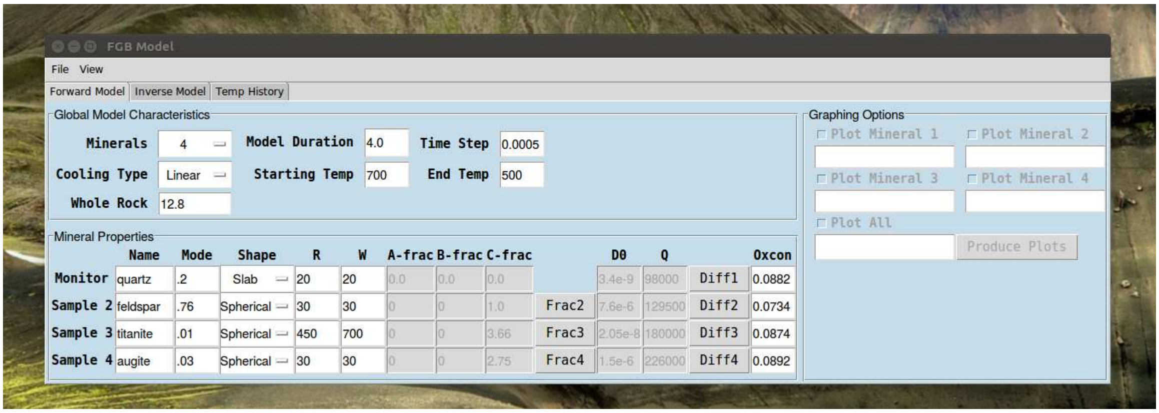
```
I have the following conversions available for mineral --> monitor
------------------------------------------------------------------

albite --> calcite
□   albite-calcite  0      0.0    -0.56   600-800    Clayton et al. (1989)         ---

calcite --> quartz
□   calcite-quartz  -0.0   -0.0   -0.38   600-1000   Clayton et al. (1989)         ---
□   calcite-quartz  -0.0   -0.0   -0.87   100-700    Sharp and Kirschner (1994)    ---
---------------------------------------------------------------
```

Minerals: 4
Cooling Type: Linear
Whole Rock: 12.8

Mineral Properties

	Name	Mode	Shape	R	W	A-frac	B-frac	C-frac		D0	Q		Oxcon
Monitor	albite		Slab			0.0	0.0	0.0				Diff1	
Sample 2	quartz		Slab						Frac2			Diff2	
Sample 3			Slab						Frac3			Diff3	
Sample 4			Slab						Frac4			Diff4	

Figure 2.7: Search menu for fractionation factors.



Global Model Characteristics

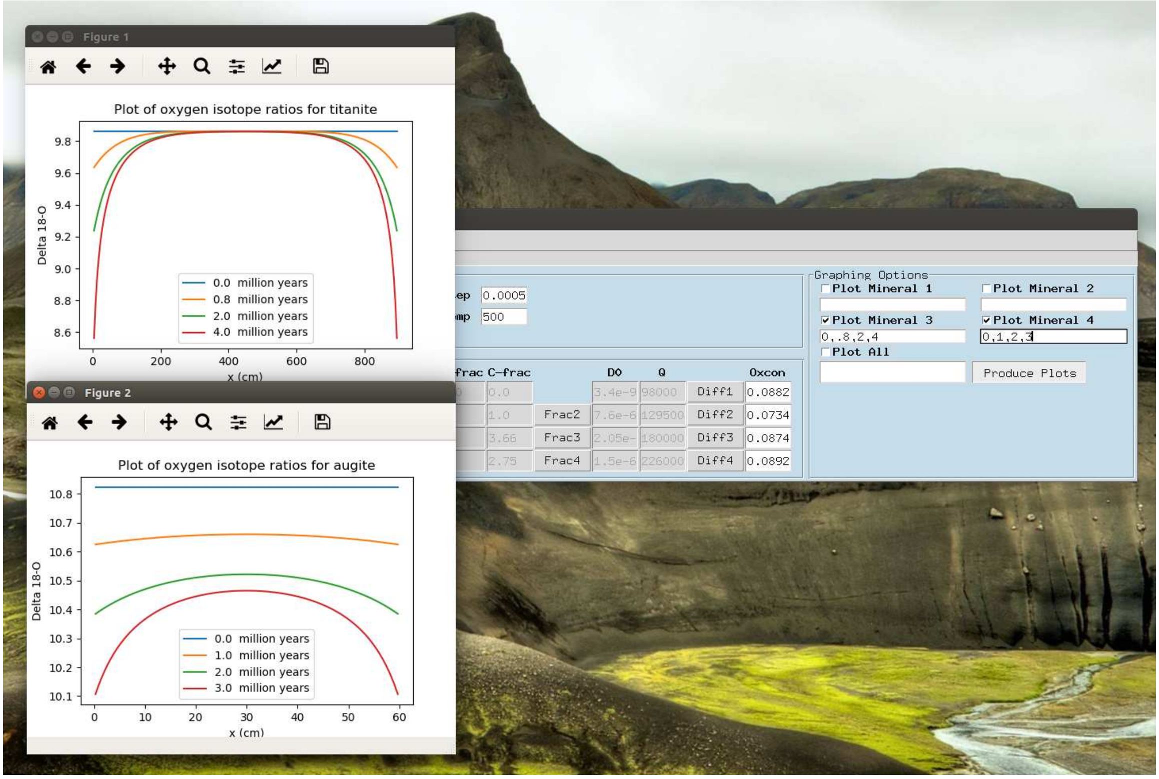
Minerals: 4
Model Duration: 4.0
Time Step: 0.0005
Cooling Type: Linear
Starting Temp: 700
End Temp: 500
Whole Rock: 12.8

Mineral Properties

	Name	Mode	Shape	R	W	A-frac	B-frac	C-frac		D0	Q		Oxcon
Monitor	quartz	.2	Slab	20	20	0.0	0.0	0.0		3.4e-9	98000	Diff1	0.0882
Sample 2	feldspar	.76	Spherical	30	30	0	0	1.0	Frac2	7.6e-6	129500	Diff2	0.0734
Sample 3	titanite	.01	Spherical	450	700	0	0	3.66	Frac3	2.05e-8	180000	Diff3	0.0874
Sample 4	augite	.03	Spherical	30	30	0	0	2.75	Frac4	1.5e-6	226000	Diff4	0.0892

Figure 2.8: Final example of model input.

'File−>Run' and the program will begin, the model simulation will take 10-30 seconds depending on your system. After the simulation is complete you can plot any minerals delta-18 values at any time, or save the entire simulation to a csv file. If you've used the parameters from importing *Examples/Ex_Params1.txt*, you should get the plots shown in Figure 2.9 by using times 0, .8, 2, 4 for mineral 3 and 0,1,2,3 for mineral 4.



ep 0.0005

mp 500

frac	C-frac		D0	Q		Oxcon
0	0.0		3.4e-9	98000	Diff1	0.0882
	1.0	Frac2	7.6e-6	129500	Diff2	0.0734
	3.66	Frac3	2.05e-	180000	Diff3	0.0874
	2.75	Frac4	1.5e-6	226000	Diff4	0.0892

Graphing Options

☐ Plot Mineral 1

☐ Plot Mineral 2

☑ Plot Mineral 3: 0,.8,2,4

☑ Plot Mineral 4: 0,1,2,3

☐ Plot All

Figure 2.9: Final example of model input.

CHAPTER 3

INVERTING THE FGB MODEL

3.1 Background

Suppose we have a forward model G, that accepts a vector of model inputs $\boldsymbol{m}$ and produces a vector of data outputs $\boldsymbol{d}$. Then the inverse problem, of finding model inputs given data outputs, can be expressed as the optimization problem (3.1). (Aster et al. (2018)).

$$\min_{\boldsymbol{m}} \ \|G(\boldsymbol{m}) - \boldsymbol{d}\|_2^2 \tag{3.1}$$

If G were assumed to be linear, then this would be a simple least squares problem and the Normal Equations could be used to analytically find the optimal $\boldsymbol{m}$. However, in the general case G may be very complex, and its inverse may have many solutions. To prevent the overfitting of noise in data, a regularization term is added to the optimization problem(3.2). The matrix L can be constructed to penalize first or second order derivatives in our model vector $\boldsymbol{m}$. These regularizations have the added benefit of collapsing many possible solutions of the inverse problem to a single best solution when the data does not entirely constrain the model (Aster et al. (2018)). We achieve an optimal balance between data fit and solution smoothness using L-curves; this will be discussed in more detail in the following section.

$$\min_{\boldsymbol{m}} \ \|G(\boldsymbol{m}) - d\|_2^2 + \beta\|L\boldsymbol{m}\|_2^2 \tag{3.2}$$

There does not exist an analytic solution for the general inverse optimization problem (nonlinear G); thus, we must employ iterative methods. A well-known method for this is the Levenberg-Marquardt (LM) algorithm. First, we define the vector valued function F according to eq. 3.3 and 3.4, where σ_i represents the uncertainty of our data measurement d_i.

$$f_i(\boldsymbol{m}) = \frac{G(\boldsymbol{m})_i - d_i}{\sigma_i} \tag{3.3}$$

$$F = \begin{bmatrix} f_1(\boldsymbol{m}) \\ f_2(\boldsymbol{m}) \\ \vdots \\ f_n(\boldsymbol{m}) \end{bmatrix} \tag{3.4}$$

The Jacobian of F is then defined in typical fashion:

$$J = \begin{bmatrix} \frac{\partial f_1(\boldsymbol{m})}{\partial m_1} & \cdots & \frac{\partial f_1(\boldsymbol{m})}{\partial m_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_m(\boldsymbol{m})}{\partial m_1} & \cdots & \frac{\partial f_m(\boldsymbol{m})}{\partial m_n} \end{bmatrix} \tag{3.5}$$

Finally, the LM algorithm starts with an initial solution and iteratively updates this solution by solving eq. 3.6 for $\Delta\boldsymbol{m}$, and then adding $\Delta\boldsymbol{m}$ to our current solution. For a more complete derivation of the algorithm, please see Aster et al. (2018).

$$\left(\boldsymbol{J}(\boldsymbol{m}^{(k)})^T\boldsymbol{J}(\boldsymbol{m}^{(k)}) + \beta^2 L^T L + \lambda I\right)\Delta m = -\boldsymbol{J}(\boldsymbol{m}^{(k)})^T\left(\boldsymbol{G}(\boldsymbol{m}^{(k)}) - d\right) - \beta^2 \boldsymbol{L}^T\boldsymbol{L}m^{(k)} \tag{3.6}$$

The parameter β is a tuning parameter that controls the inverse solver's priority for model smoothness. The parameter λ is used to ensure convergence and is updated at each iteration depending on whether or not the solution is getting better. The LM algorithm has the advantage of converging where gradient descent would, but also has quadratic convergence under appropriate assumptions (Aster et al. (2018)).

3.2 Sensitivity Analysis of Measurement Uncertainties

For any forward/inverse problem pair, there exists uncertainties in both the model parameters and the recorded data. This feature is captured in equations 3.7 and 3.8. For inverse problems it is assumed that the forward model accurately captures the physics of the process, thus $\boldsymbol{G}(\boldsymbol{m}_{\text{true}}) = \boldsymbol{d}_{\text{true}}$, and $\boldsymbol{d}_{\text{noise}}$ is noise generated purely by measurement error and is not correlated with $\boldsymbol{m}_{\text{noise}}$ or $\boldsymbol{G}$ in anyway.

$$\boldsymbol{m}_{\text{measured}} = \boldsymbol{m}_{\text{true}} + \boldsymbol{m}_{\text{noise}} \tag{3.7}$$

$$\boldsymbol{d}_{\text{measured}} = \boldsymbol{d}_{\text{true}} + \boldsymbol{d}_{\text{noise}} \tag{3.8}$$

Define $\boldsymbol{\xi}$ as the effect of our measurement error on the model output in the following fashion:

$$\boldsymbol{\xi} = \boldsymbol{G}(\boldsymbol{m}_{\text{measured}}) - \boldsymbol{G}(\boldsymbol{m}_{\text{true}}) \tag{3.9}$$

Recall that G is not assumed to be linear, so this expression cannot be simplified. An expression for the $\boldsymbol{\sigma}$ weights used in 3.3 can now be derived.

$$
\begin{aligned}
\sigma_i^2 &= \mathrm{Var}\left[G(\boldsymbol{m}_{\text{measured}})_i - (d_{\text{measured}})_i\right] && (3.10)\\
\sigma_i^2 &= \mathrm{Var}\left[G(\boldsymbol{m}_{\text{true}})_i + \xi_i - (d_{\text{true}} + d_{\text{noise}})_i\right] && (3.11)\\
\sigma_i^2 &= \mathrm{Var}\left[G(\boldsymbol{m}_{\text{true}})_i - (d_{\text{true}})_i + \xi_i - (d_{\text{noise}})_i\right] && (3.12)\\
\sigma_i^2 &= \mathrm{Var}\left[\xi_i - (d_{\text{noise}})_i\right] && (3.13)\\
\sigma_i^2 &= \mathrm{Var}\left[\xi_i\right] + \mathrm{Var}\left[(d_{\text{noise}})_i\right] && (3.14)
\end{aligned}
$$

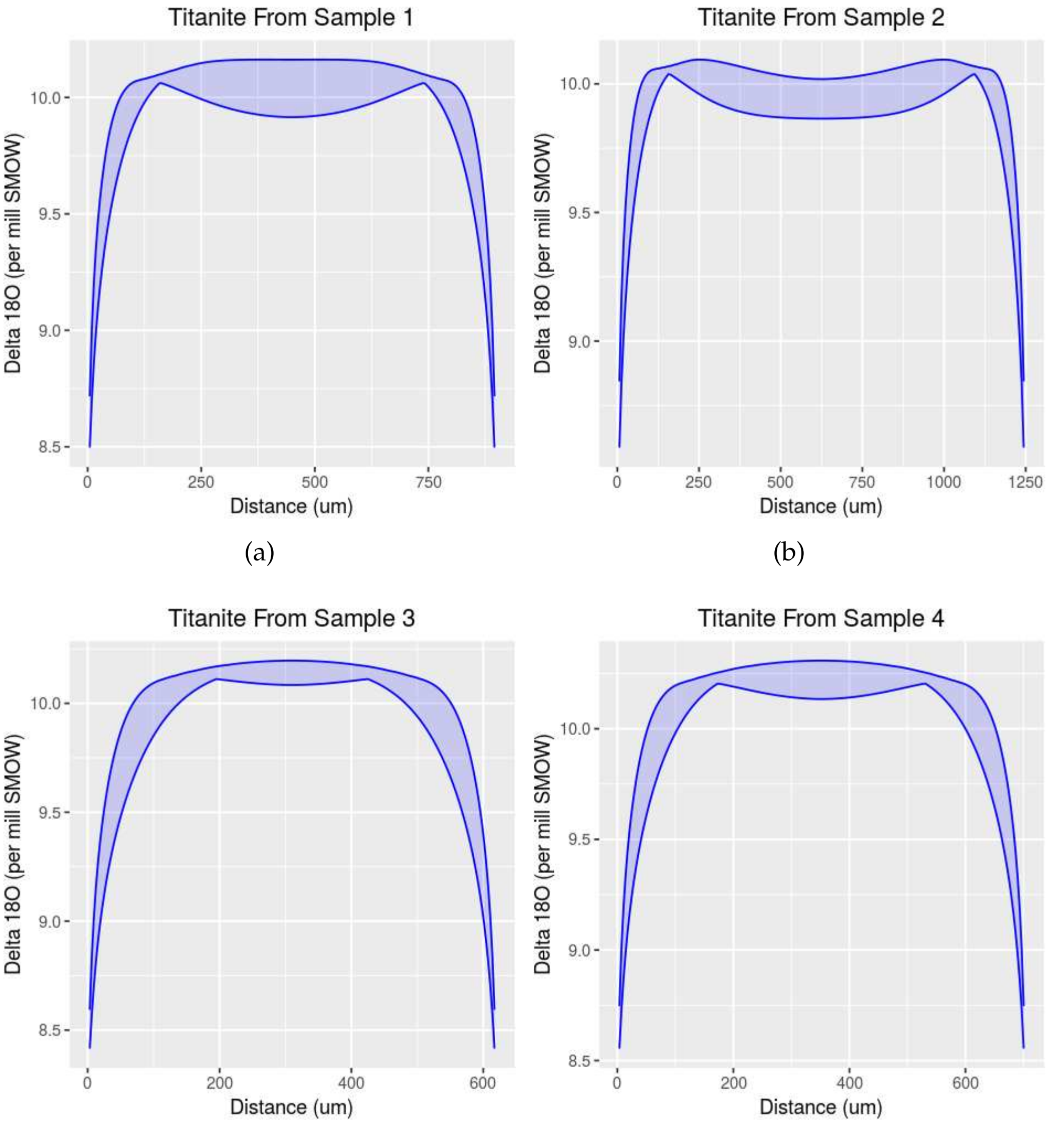
The variance of $\boldsymbol{d}_{\text{noise}}$ comes from the analytical uncertainty of the mass spectrometer used to measure the oxygen-isotope ratios. The variance of $\boldsymbol{\xi}$ as defined by 3.9, has to be estimated by repetitive forward model runs. This section focuses on estimating the effects of the uncertainties associated with the mineral diffusivities on model output.

All the samples, real and synthetic, analyzed in this research contain four minerals that are simultaneously simulated. For each mineral, the forward model accepts two diffusivity parameters, D_0 and Q, which are used to calculate the diffusivity by the Arrhenius relation:

$$D = D_0 \exp\left(-Q/RT\right) \qquad (3.15)$$

where R is the gas constant and T is the current temperature. This gives us 8 real-valued inputs h_i, with 8 associated uncertainties $\sigma(h_i)$. For each input, we try the three possibilities: $h_i - \sigma(h_i)$, h_i, and $h_i + \sigma(h_i)$. This results in $3^8 = 6561$ possible forward runs; with the speed-ups achieved by the C code this only takes a few minutes. The minimum and maximum values for each model output, taken over all possible forward runs, are kept. The minimum and maximum values are then considered the range of reasonable estimates for our model output, with the difference between them acting as an estimate for output variance.

An example of the model output uncertainties is shown in Figure 3.1. The ranges of model output can be seen to tighten near the mineral grain boundaries. This is a desirable feature, as it will be shown in chapter 4 that the data points nearest the grain boundary contain the most useful information for thermal history inversion.



(a)

(b)

(c)

(d)

Figure 3.1: Example of model output uncertainties produced by diffusivity uncertainties. The upper and lower bounds, indicated by the solid blue lines, are computed from the point-wise maximum and minimum outputs from the FGB algorithm over all possible combinations of 2SD diffusivity parameter uncertainties.

3.3 Applying the Levenberg-Marquardt Algorithm in Python and C

With all other model parameters known, the inverse solver attempts to recover the time-temperature history of a mineral sample from the diffusion profiles of the grains within that sample. This works by first descretizing the time-temperature history as shown in Figure 3.2; where the solution is assumed to be a linear interpolator between a finite number of points equally spaced apart. We can then use the LM algorithm as outlined in section 1.3, where the vector of temperatures is considered the model input vector $\boldsymbol{m}$.

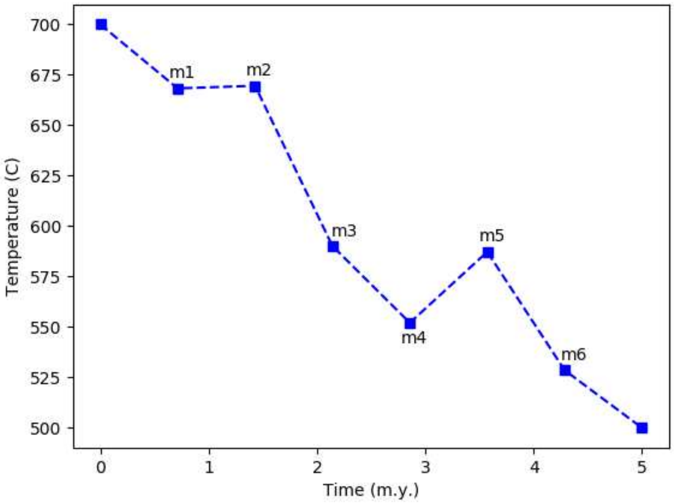


Figure 3.2: Example of the discretized model input. The actual input is a vector of temperatures, $\boldsymbol{m}$, corresponding to the temperature at each of the equally spaced time-periods. A linear interpolator is then used to create a continuous time-temperature history.

Because the LM algorithm is an iterative method, it requires an initial solution. The user provides this in the form of a csv file, with times in the first column and corresponding temperatures in the second column. Only values in the second column are changed; at each step they are updated by solving 3.6. At each iteration, if the error improves, λ is multiplied by a factor of $1/8$. Otherwise, it is doubled. The λ term is used to ensure convergence when the condition numbers of our matrices become large as is often the case with quadratic methods and poor initial guesses.(Aster et al. (2018)). The program is built to accept any number of data files corresponding to the diffusive profiles for each mineral in the sample. Additionally, if multiple samples are taken from the same region and assumed to have experienced identical thermal conditions, the program allows these to be combined to more accurately recover the shared thermal history.

To select the optimal β that balances data fit and model smoothness, we compute L-curves. This works by running the inverse solver with progressively larger values of β, and then plotting the error $||G(\boldsymbol{m}) - d||_2^2$ vs. the solution smoothness $||Lm||_2^2$. This produces plots similar to that shown in Figure 3.3. We then select the β associated with the point found in the kink of the L-curve. Ad-

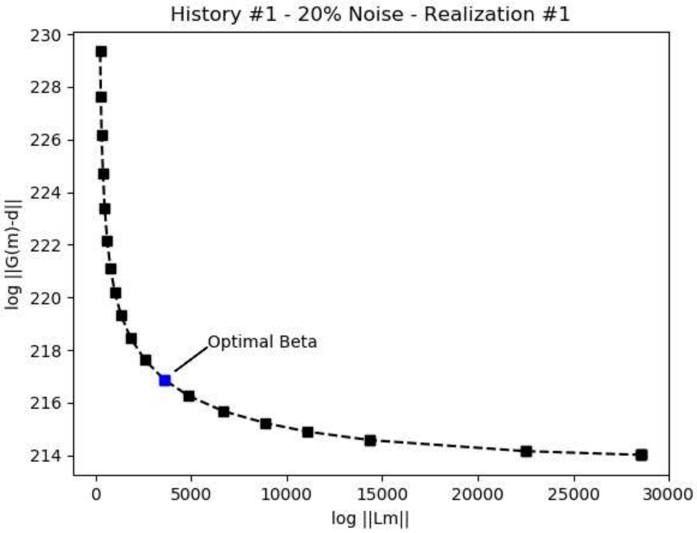


Figure 3.3: Example of L-curve used to choose optimal smoothing parameter.

ditionally, to ensure that our solutions are actual global minimums and not just local entrapments, we use the multi-start approach. This works by simply rerunning the inverse solver with many different initial solutions, and then comparing all the final solution errors. If there are many equally good solutions (similar errors), then we compare the structure of these solutions to find general trends and make statements about persistent features, such as reheating events.

3.4 Evaluation of Inverse Capabilities on Simulated Data

Before applying the inverse algorithm to real world data, it is evaluated on simulated data to determine what features it is capable of recovering. The overall process is to start with a known time-temperature history, run the forward model, keep only a sample of the data points from the output, add noise to these data points, and then run the inverse solver on this noisy data. We do this for many different temperature histories. The output that is kept and fed into the inverse solver is shown Figure 3.4, where $G(\boldsymbol{m})_i$ is the actual output and d_i is the value stored that has been obscured with noise. The level of noise is chosen to match the real data we have.

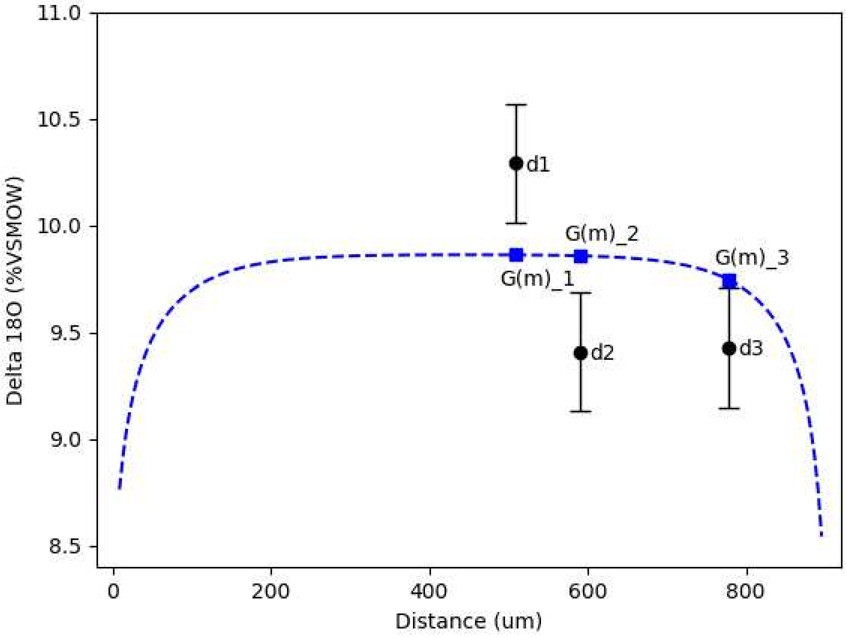


Figure 3.4: Example of the stored data from simulated cases.

3.4.1 Analysis of First Synthetic Sample

For the first test case we start with the thermal history shown in Figure 3.5. After running the forward model and adding noise to the points that are kept, we get the data shown in 3.6. The data kept are chosen to match the locations of our real world isotope measurements. The data in Figure 3.6 is the only information that the inverse solver has access to. We run the inverse solver with 17 different initial solutions, some of which are shown in Figure 3.7. Then we take the local minimum that these starting solutions fall into and plot the resulting error of each on the bar plot shown in Figure 3.8. The plot is an overlaid bar plot with the regularization shown on top (solution roughness $||Lm||$). We can see how little area of the bar plot is made up of the regularization term. We deliberately used a low β value for this first case to gauge how well the data constrains the problem. The total errors on the bar plot corresponding to each solution are all roughly equal, showing that our data does not entirely constrain our solution.

The solutions are then split into three separate groups based on their visual similarities. The solutions are shown in the right of Figures 3.9 through 3.10, with their corresponding errors highlighted on the left. We can see that group A recreates the reheating event slightly early, and counteracts this by proceeding with a cooler history than the actual solution. Group C does the opposite, recreating the history late and preceding by a cooler region. However, every solution

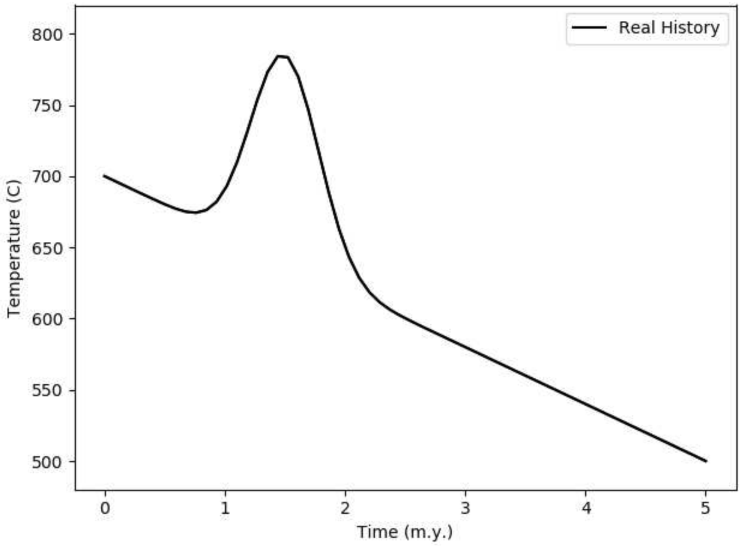


Figure 3.5: Time-temperature history of our first synthetic test case.

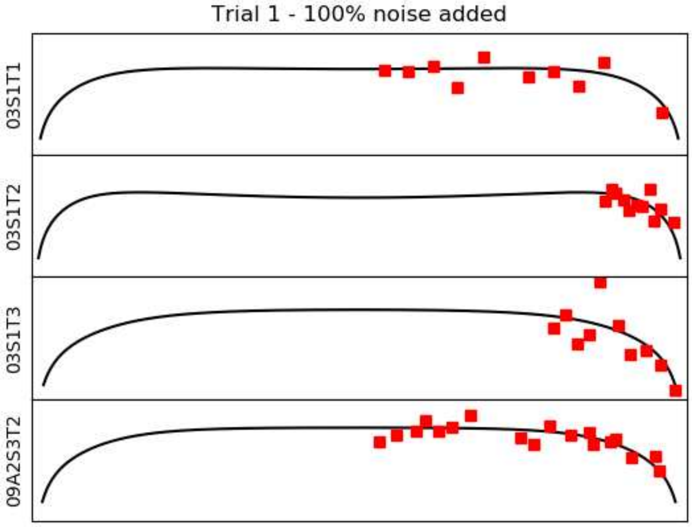


Figure 3.6: Simulated data of first test case.

correctly recovers both the height and duration of this reheating event. To find the optimal smoothing parameter that collapses these many possible solutions we compute an L-curve. This is shown in Figure 3.12, where the point highlighted in red is the current value we used to generate plots 3.9 through 3.11. If we increase the smoothness to that highlighted in blue, we condense our wide

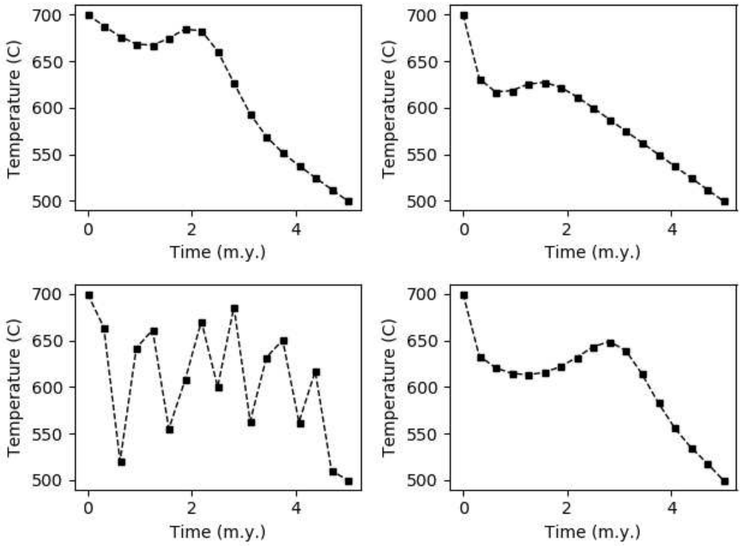


Figure 3.7: Four of the 17 initial solutions used to begin the inverse solver.

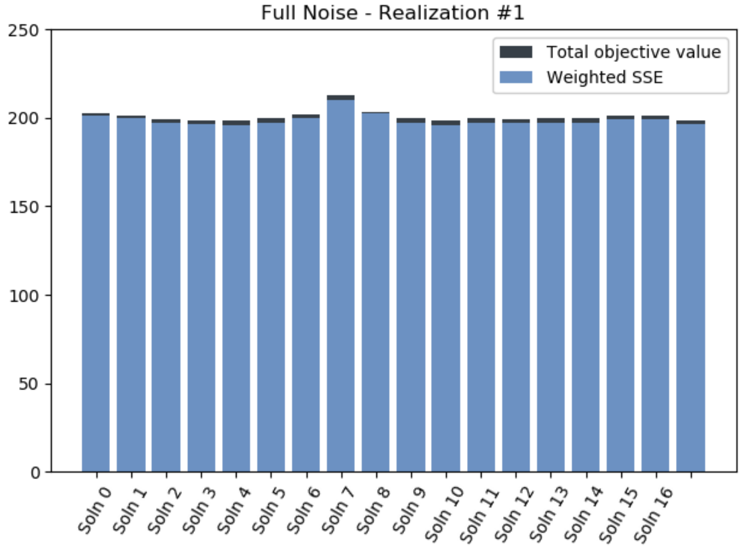
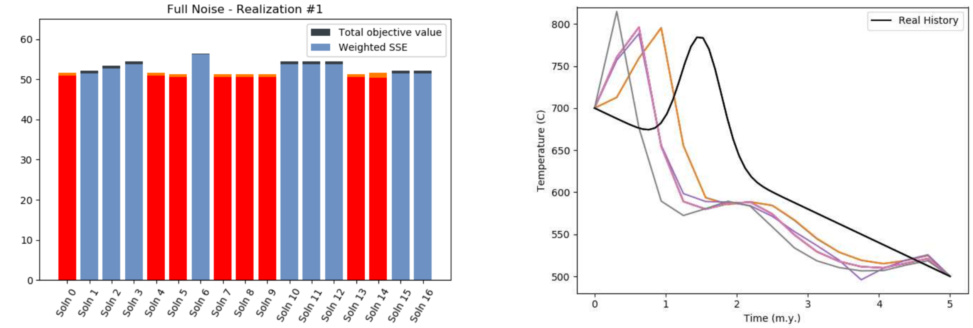


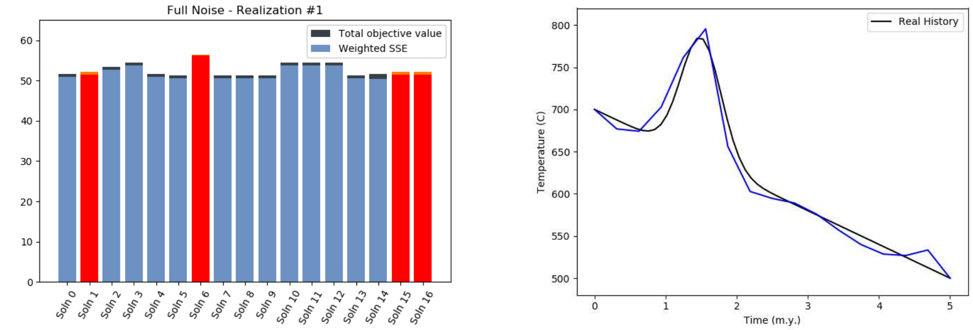
Figure 3.8: Bar plot of errors from the 17 local minimums achieved from our inverse solver.

spread of solutions to a single best solution to that shown in 3.13. However, this has significantly smoothed out the reheating event.



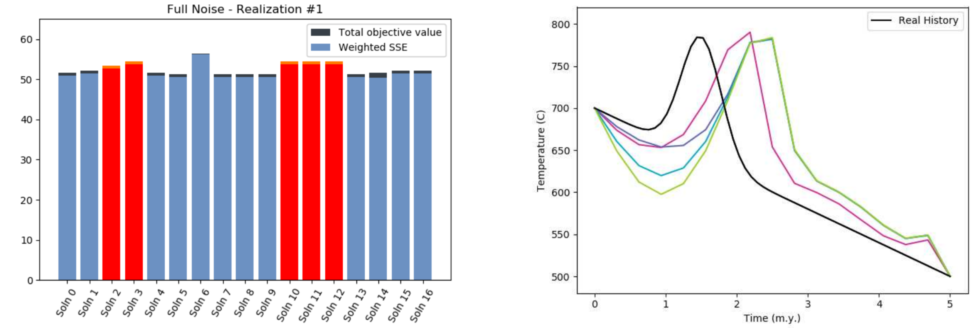
(a) Highlighted Errors (b) Actual Solutions

Figure 3.9: Grouped solutions to the first synthetic test case (group A).



(a) Highlighted Errors (b) Actual Solutions

Figure 3.10: Grouped solutions to the first synthetic test case (group B).



(a) Highlighted Errors (b) Actual Solutions

Figure 3.11: Grouped solutions to the first synthetic test case (group C).

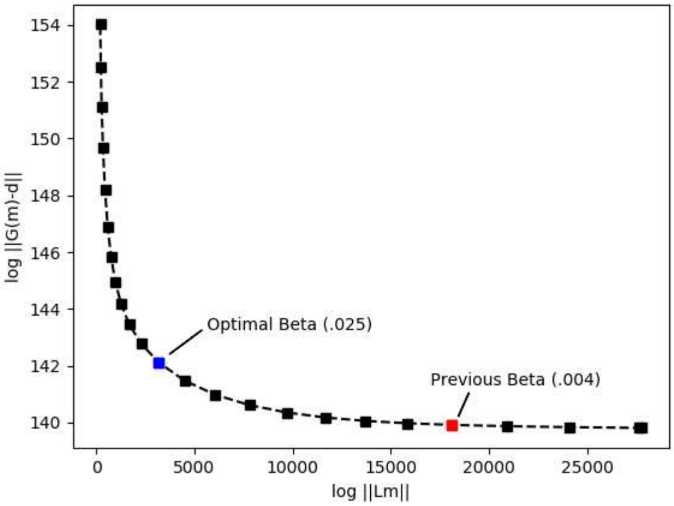


Figure 3.12: L-curve with previous and optimal smoothing parameter values highlighted.

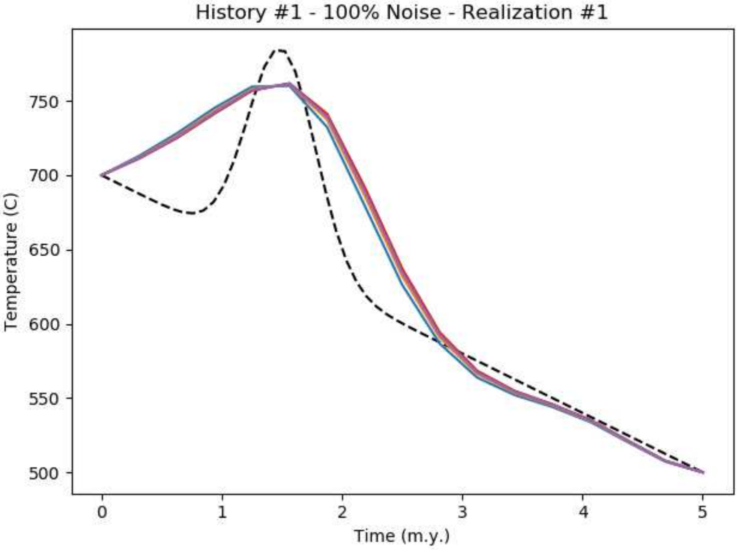


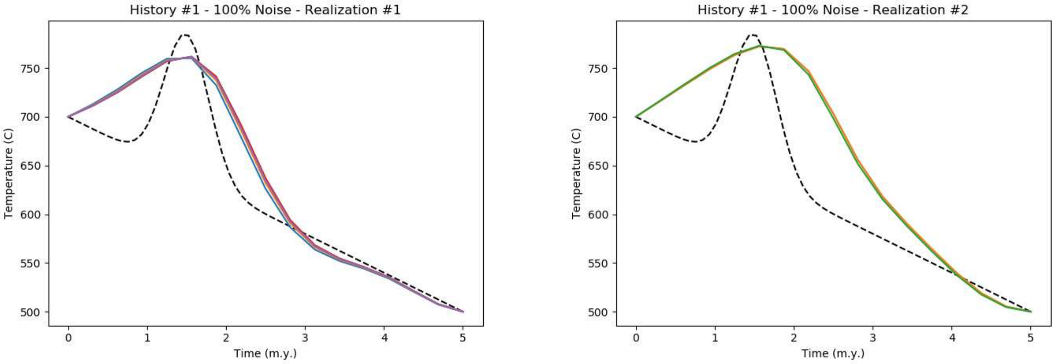
Figure 3.13: Collapsed solutions from first test case after regularization has been turned up to optimal value.

3.4.2 Summary of Inversion Results From All Simulated Cases

We want to create a catalog of what features the inverse solver does and does not pick up. Reheating events may not be recoverable if they occur too early or late and this needs to be examined before drawing any conclusions from our real world data. A disadvantage of the LM method is that it does not allow the x values (times) to change, only the y values (temperature). Alternative methods such as Markov Chain Monte Carlo used by Gallagher (2011), allow data points to be created, eliminated, and shifted as need. Our solver works as a proof-of-concept to the amount of thermal information stored inside the mineral grains diffusive profiles.

By looking at the thermal history inversions of three different reheating events using synthetic data with the full level of expected noise (Fig. 3.14 through 3.16), it can be seen that our inverse solver cannot with any degree of confidence state when the reheating event occurred. However, comparing these recovered solutions to those found in 3.20, we can see that our solver can clearly distinguish between thermal histories which contain reheating events and thermal histories of a simple linear form. If we then reduce the level of noise to 20% of what our current real world samples contain, in Figures 3.17 through 3.19 it can be seen that the inverse solver is now able to correctly determine the time at which the reheating even occurred to within a million years. Additionally, we see that the closer a reheating event occurs to the end of our cooling period the more the solution becomes smoothed over by the regularization.

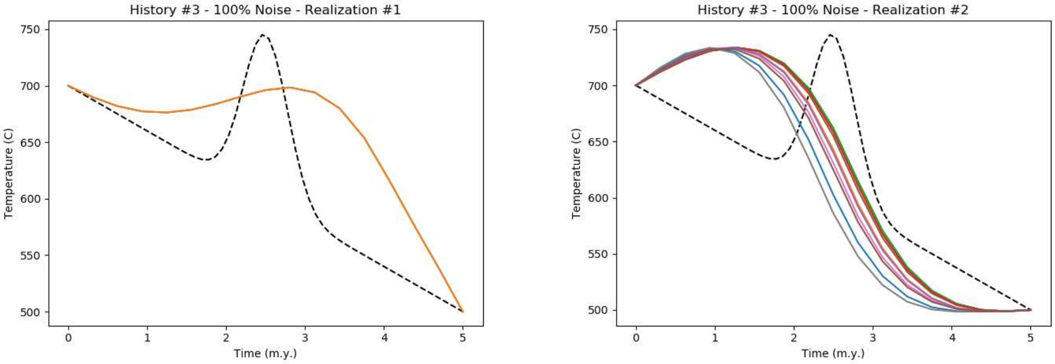
All the synthetic data used in this chapter were 'measured' at the same locations as our real world data and with similar levels of noise. This was done to provide context when analyzing the inverse solutions from our real world data, so that we have an appropriate representation of the solver's current ability. We will explore how the actual sample locations affect the inverse solutions in more detail in chapter 4.



(a)

(b)

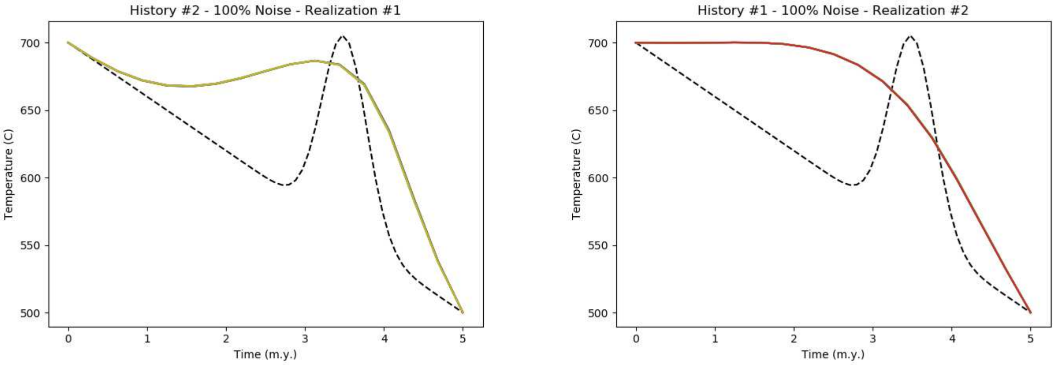
Figure 3.14: Inversion of an early reheating event using full noise.



(a)

(b)

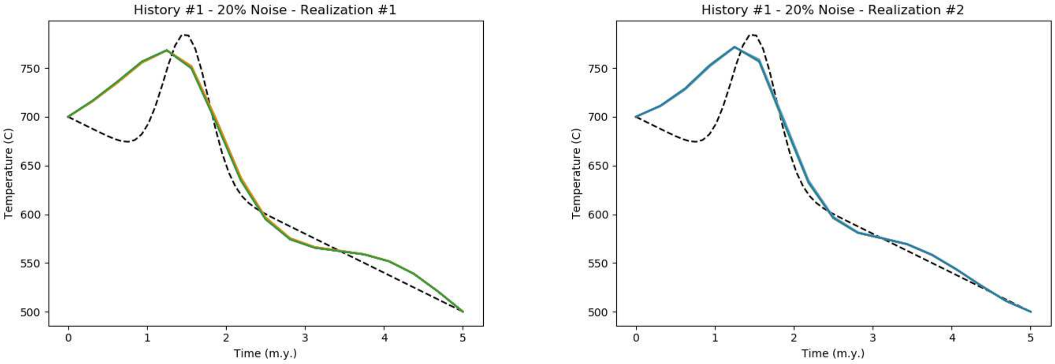
Figure 3.15: Inversion of a mid-history reheating event using full noise.



(a)

(b)

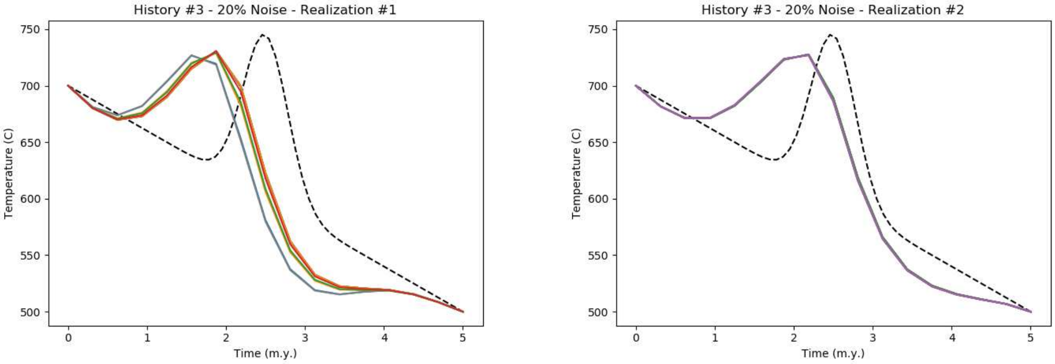
Figure 3.16: Inversion of a late reheating event using full noise.



(a)

(b)

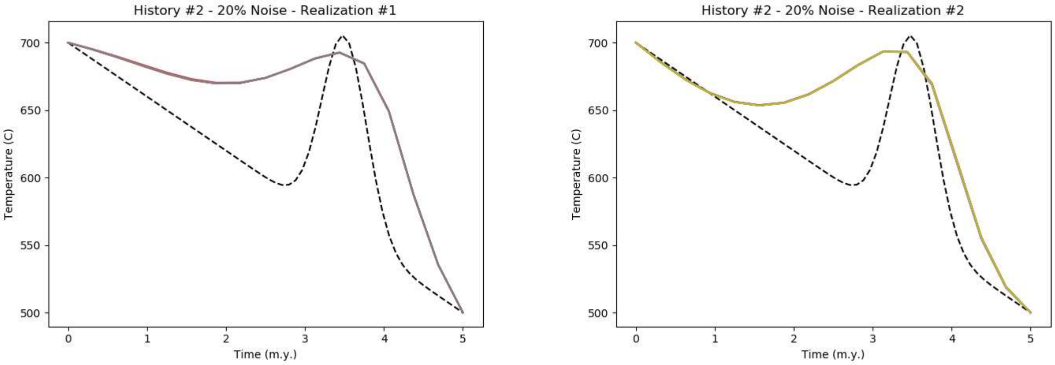
Figure 3.17: Inversion of an early reheating event using low noise.



(a)

(b)

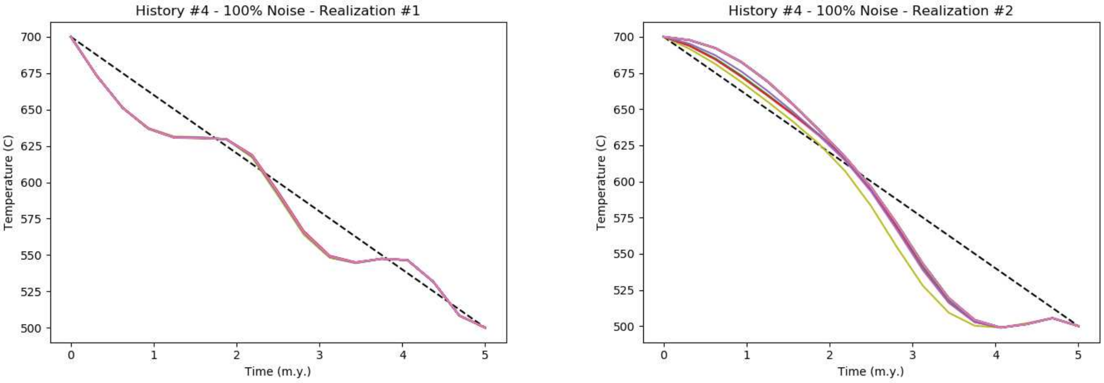
Figure 3.18: Inversion of a mid-history reheating event using low noise.



(a)

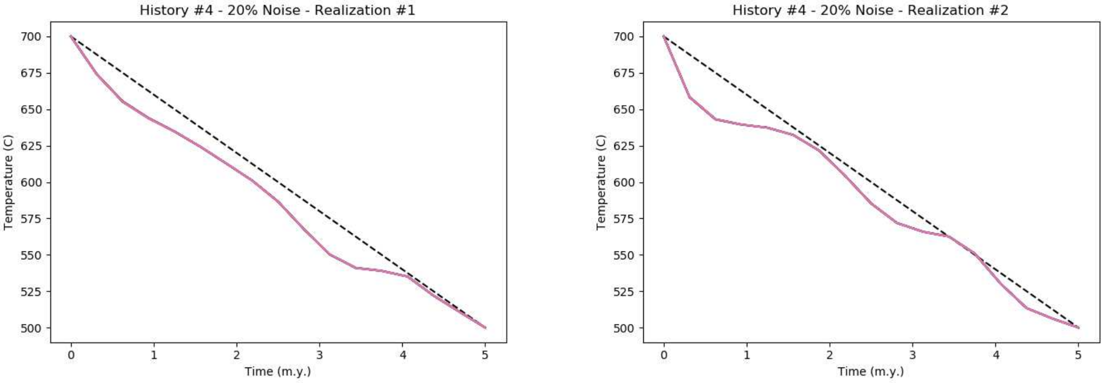
(b)

Figure 3.19: Inversion of a late reheating event using low noise.



(a) (b)

Figure 3.20: Inversion of a linear profile using full noise.



(a) (b)

Figure 3.21: Inversion of a linear profile using low noise.

3.5 Alternate Methods of Regularization

As was seen in the figures from the previous section, the curvature regularization over-smoothed reheating events when they occurred closer to the end of our cooling period. This is because the temperatures are lower near the end and diffusion is greatly slowed down. This means that reheating events that occur during cooler periods of time are not well-recovered with the curvature regularization. And without smoothing the solutions were not physically reasonable. Thus, we wish to try alternate forms of regularization. Since the dominant mode of cooling is expected to be linear in the absence of other physical phenomena, we will regularize by penalizing deviation from linearity. This means portions of the cooling history will only be changed from linearity in the presence of strong evidence. The new objective function that we wish to optimize over is now given by eq. 3.16, where $\boldsymbol{m}_{linear}$ is the vector of model parameters that describes a linear cooling trend over the specified time-period.

$$\|\boldsymbol{G}(\boldsymbol{m})-\boldsymbol{d}\|_2^2+\beta\|\boldsymbol{m}-\boldsymbol{m}_{linear}\|_2^2 \tag{3.16}$$

To find local minimum we use the same iterative approach with multi-starts, except the model updates are now made by:

$$\left(\boldsymbol{J}(\boldsymbol{m}^{(k)})^T\boldsymbol{J}(\boldsymbol{m}^{(k)})+\beta^2\boldsymbol{I}\right)\Delta m=-\boldsymbol{J}(\boldsymbol{m}^{(k)})^T\left(\boldsymbol{G}(\boldsymbol{m}^{(k)})-d\right)-\beta^2(\boldsymbol{m}^{(k)}-\boldsymbol{m}_{linear}) \tag{3.17}$$

Note that the $\beta\boldsymbol{I}$ term is different than in eq. 3.6. In eq. 3.6 the $\boldsymbol{I}$ term is used to stabilize the iterations and does not effect the final solution achieved. In this equation, it is used as a regularization, which is why the iterative steps have an associated term on the right. β is a tuning parameter. Using this deviation from linearity regularization and selecting the optimal tuning parameter we get the spread of inverse solutions shown in Figure 3.22. This offers a dramatic improvement over the solutions achieved in Figures 3.16 and 3.19. Thus, deviation from linearity regularization should be used for time-temperature history recovery during cooler periods.

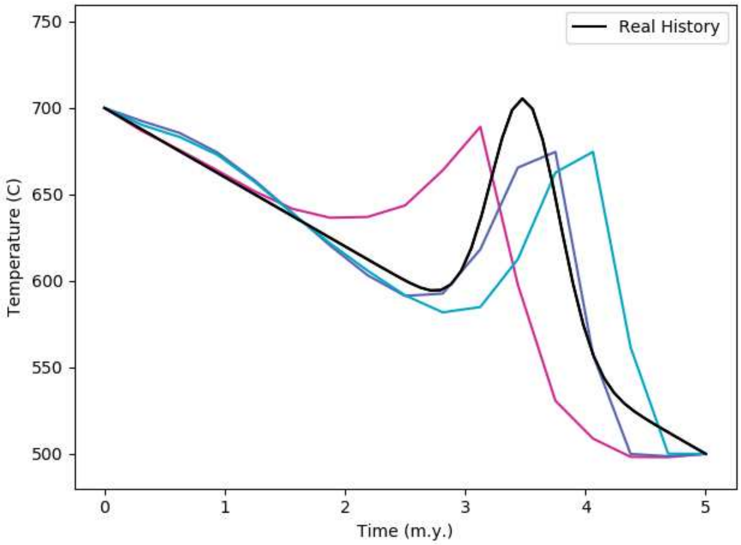
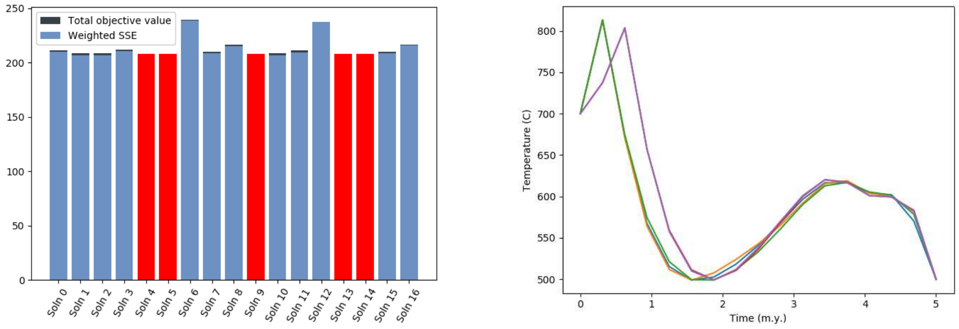


Figure 3.22: Multiple equally-good inverse solutions using deviation from linearity regularization for late reheating event.

Combinations of both curvature and deviation from linearity can be used as regularization. However, with two tuning parameters there is no longer a simple criteria by which we can select optimal parameter pairs as we can with the L-curve for one tuning parameter. Multiple combinations were tried, but no successful values were found that produced better solutions than the individual regularizations by themselves. This is a potential avenue for future research in which deep exploration could be performed on combination pairs.

3.6 Inverse Results from Titanite Samples

We are very limited in the amount of real world data available to us. However, with the software package completed, future data will be easily analyzed as it is recorded. The data we do have available is four samples all collected from within the same region. Thus, we can use the data together as the minerals should have undergone identical cooling conditions. Within each sample there are four abundant minerals: titanite, feldspar, quartz, and augite. For each sample, we have roughly 25 data points collected along a single side of a single grain of titanite within the sample. This results in 102 data points used to reconstruct the discretized 17-point time-temperature history. Running the inverse solver results in the solutions shown in the right of Figures 3.23 through 3.26, with corresponding errors shown in the bar plots on the left.



(a) Highlighted Errors

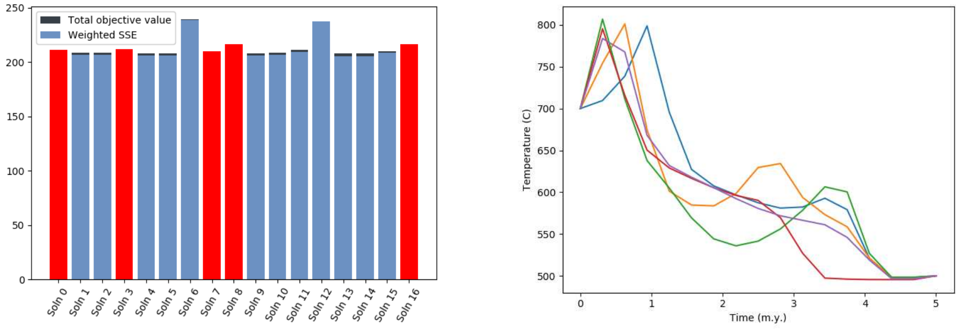
(b) Actual Solutions

Figure 3.23: Grouped solutions to real titanite samples (group A).

While there appears to be a reheating event in some of the solutions, there are other equally good solutions that do not show one. Thus, we cannot say whether or not there was a reheating event. What these solutions do indicate, is that 700^{o}C is likely an underestimate of the peak temperature conditions attained in this region. With more data from different minerals and grain sizes we would likely be able to better constrain the thermal history.

3.7 Using the Python GUI - Inverse Solver

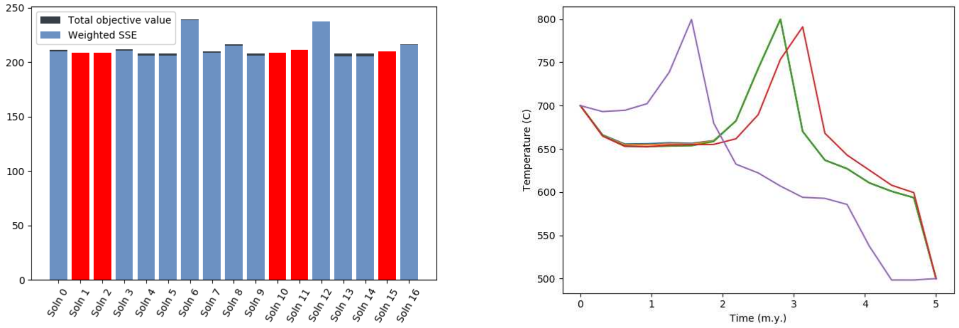
The inverse solver will accept 1-5 files containing saved model parameters created in the forward model tab (see section 2.5). Each file corresponds



(a) Highlighted Errors

(b) Actual Solutions

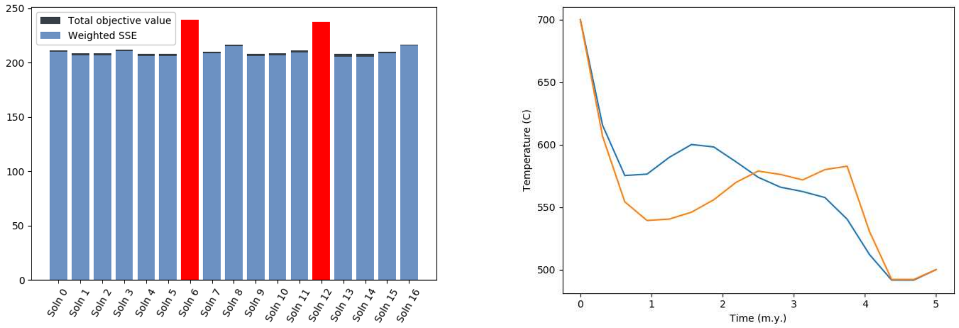
Figure 3.24: Grouped solutions to real titanite samples (group B).



(a) Highlighted Errors

(b) Actual Solutions

Figure 3.25: Grouped solutions to real titanite samples (group C).



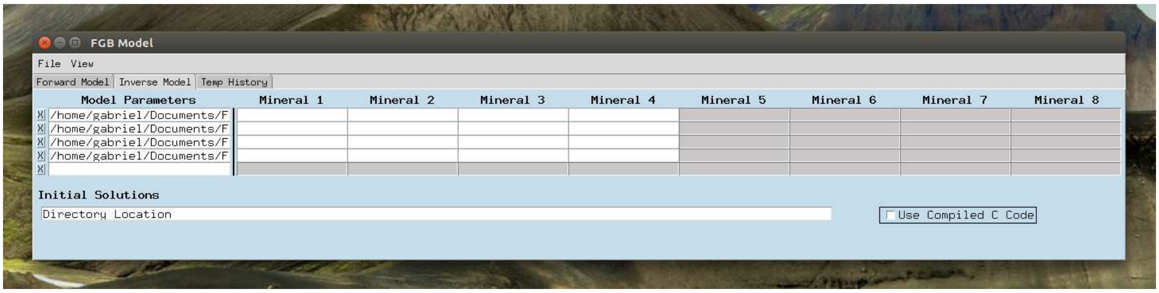
(a) Highlighted Errors

(b) Actual Solutions

Figure 3.26: Grouped solutions to real titanite samples (group D).

to a single mineral sample. For each sample, and each mineral grain in that sample, we can then upload a data file containing an isotopic profile and associated uncertainties. The files referenced can be download from the repository https://github.com/gkropf/FastGrainBoundary-DiffusionSolver.

By clicking on the rows of section 'Model Parameters' in the tab 'Inverse Model', we can upload all four of our example of model parameters: *Ex_Params1.txt*, *Ex_Params2.txt*, *Ex_Params3.txt*, and *Ex_Params4.txt*. This will result in the screen:



	Model Parameters	Mineral 1	Mineral 2	Mineral 3	Mineral 4	Mineral 5	Mineral 6	Mineral 7	Mineral 8
☒	/home/gabriel/Documents/F								
☒	/home/gabriel/Documents/F								
☒	/home/gabriel/Documents/F								
☒	/home/gabriel/Documents/F								
☒									

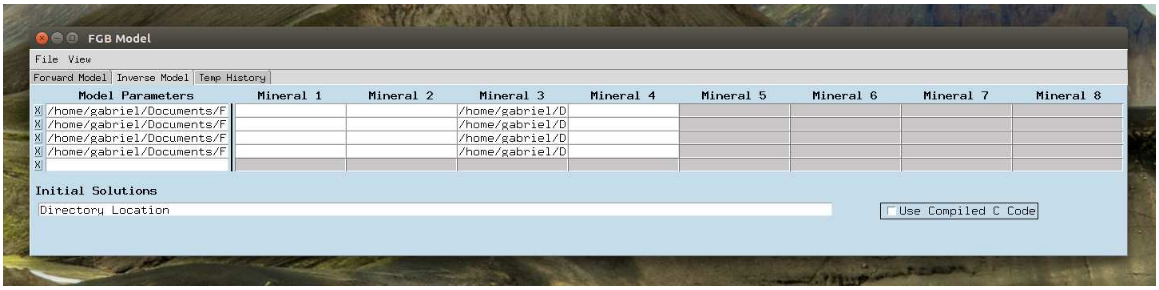
Initial Solutions

Directory Location

□ Use Compiled C Code

Figure 3.27: Selecting saved forward model parameter files.

Each one of these parameter files is setup to simulate the diffusive exchange between four minerals: quartz, feldspar, titanite, and augite. However, for each one we only have data available for the third mineral, titanite. Thus, we will upload our example data *Ex_Data1.txt* through *Ex_Data4.txt*, in the top four rows of the column 'Mineral 3.'



	Model Parameters	Mineral 1	Mineral 2	Mineral 3	Mineral 4	Mineral 5	Mineral 6	Mineral 7	Mineral 8
☒	/home/gabriel/Documents/F			/home/gabriel/D					
☒	/home/gabriel/Documents/F			/home/gabriel/D					
☒	/home/gabriel/Documents/F			/home/gabriel/D					
☒	/home/gabriel/Documents/F			/home/gabriel/D					
☒									

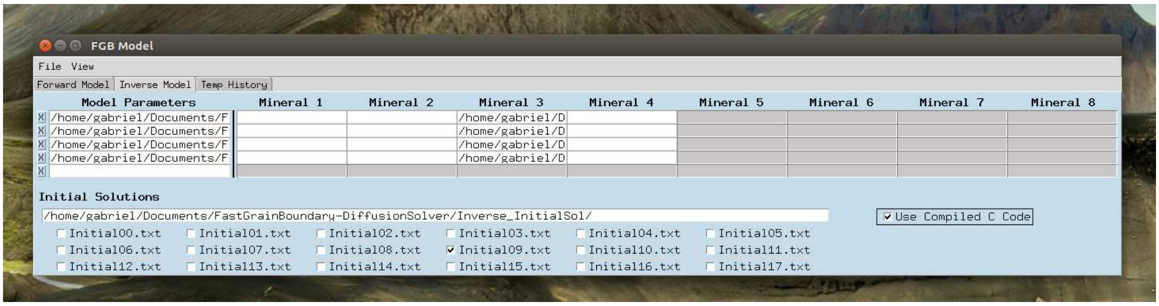
Initial Solutions

Directory Location

□ Use Compiled C Code

Figure 3.28: Selecting output data files.

Next, we will need to select which starting solutions we wish to use for our multi-start Levenberg-Marquardt algorithm. Click on the 'directory location' file dialog and select the *Inverse_InitialSol* folder. We will then elect to use *Initial09.txt* as our starting solution.



Model Parameters	Mineral 1	Mineral 2	Mineral 3	Mineral 4	Mineral 5	Mineral 6	Mineral 7	Mineral 8
☒ /home/gabriel/Documents/F			/home/gabriel/D					
☒ /home/gabriel/Documents/F			/home/gabriel/D					
☒ /home/gabriel/Documents/F			/home/gabriel/D					
☒ /home/gabriel/Documents/F			/home/gabriel/D					
☒								

Initial Solutions

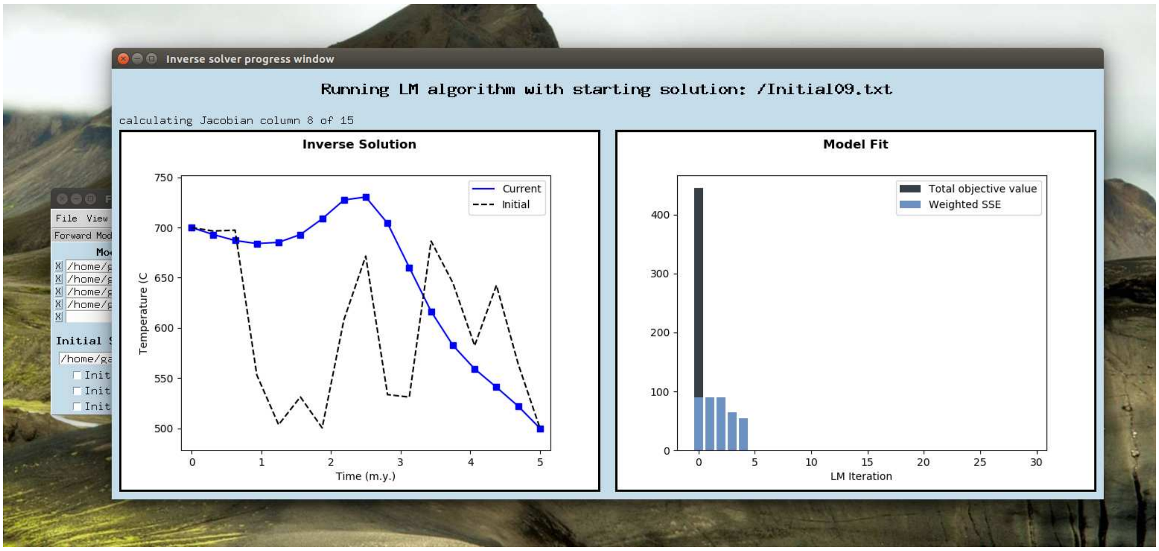
/home/gabriel/Documents/FastGrainBoundary-DiffusionSolver/Inverse_InitialSol/

☑ Use Compiled C Code

□ Initial00.txt □ Initial01.txt □ Initial02.txt □ Initial03.txt □ Initial04.txt □ Initial05.txt
□ Initial06.txt □ Initial07.txt □ Initial08.txt ☑ Initial09.txt □ Initial10.txt □ Initial11.txt
□ Initial12.txt □ Initial13.txt □ Initial14.txt □ Initial15.txt □ Initial16.txt □ Initial17.txt

Figure 3.29: Selecting initial profiles used in inverse solver.

Finally, we will click File->Run. At each iteration of the process, the left panel will show the initial solution in a black dashed line, and the current solution will be shown in blue. All the past objective values (measure of data fit and smoothness) will be shown in the bar plot on the right.



Running LM algorithm with starting solution: /Initial09.txt

calculating Jacobian column 8 of 15

Figure 3.30: Example of inverse solver progress plots.

CHAPTER 4

MEASUREMENT SCHEMES FOR IMPROVED INVERSE THERMAL HISTORY RECOVERY

We want to determine where in the diffusive isotope profile the most thermal information is stored. We cannot rely on minimizing the estimated model covariance matrix because we have employed regularization and regularization always introduces bias. Therefore, we will try eight different schemes for sample measurements with the current achievable minimum distance between successive pits, and the current values for isotope ratio uncertainties. Then we will try sample schemes with a hypothetical reduction in uncertainty and minimum distance between successive measurements. The general problem set up is the same as in section 3.4: forward model, select data points from the locations shown in the scheme illustration, add noise, and then run the inverse solver. The smoothing parameter used is individually determined for each noise realization so that the solutions from every individual noise realization are representative of what might be uncovered if we did not know the original profile.

For sample schemes 1 through 4, illustrated in Figures 4.1, 4.3, 4.5, and 4.7, measurements are taken 12 microns apart and noise is generated from a normal distribution with a standard deviation of .28‰. For each sample scheme, four separate noise realizations are generated. Then for each individual noise realization, all the equally good fitting solutions found by the inverse solver are overlaid on a plot of the actual cooling history used to forward model the data. For sample scheme 5, the the measurements are taken only 6 microns apart. For scheme 6, the noise has a standard deviation reduced to 0.06‰. For schemes 7 and 8, the measurements are taken only 6 microns apart and the noise level is reduced to ±0.06‰.

The first measurement scheme takes 12 measurements on a single side of the oxygen isotope profile. From the inverse solutions using the first measurement scheme, illustrated in Figure 4.2, the solver is able to identify the relative height and duration of our reheating event. However, for noise realizations (b) and (c), the inverse solver does not recover the linear cooling trend following the reheating event. The l_2-norm regularization can both smooth out the original reheating event or introduce an artificial second one. The second sample scheme takes six measurements on each side of the diffusive profile. Because the isotope profile is symmetric, this effectively cuts the number of measurements in half but reduces their uncertainty by a factor of $\sqrt{2}$. As seen in Figure 4.4, this scheme is slightly worse than scheme 1 because a much higher smoothing parameter is needed to collapse the solutions. Thus the reheating events are more sharply smoothed and none of the noise realizations result in solutions that correctly recover the linear sections of the cooling history. For sample scheme 3, 12 measurements are taken 12 microns apart at the center of the isotope profile. While the inverse solutions, illustrated in Figure 4.6, are highly consistent between the four noise realizations and the peak temperature of recovered solutions matches the original reheating event, the duration of the reheating event is completely lost and no portion of the linear cooling trend is correctly recovered.

For sample scheme 4, we combine 12 measurements taken from one side of the grain boundary *and* 12 measurements from the center of the grain. The cooling histories recovered using this scheme, illustrated in Figure 4.8, show no significant improvement over the solutions found using scheme 1. Thus, the thermal information recorded in the center of the grain is redundant and measurements should be taken closest to the grain boundary in order to improve the time-temperature history recovery. This is not unexpected, as the current temperature primarily influences the equilibration of the grain boundaries, and it takes time for this thermal information to diffuse to the center of the grain.

For sample scheme 5, we take 24 measurements 6 microns apart at the edge of the grain boundaries. This is a theoretical reduction of the minimum spacing between successive measurements by a factor of 2. For sample scheme 6, we use the same measurement locations as scheme 1 but reduce the noise level by a factor of 5. We can see from the inverse solutions in Figures 4.10 and 4.12, that both measurement improvements offer noticeably better inverse solutions than those obtained in 4.2. However, the inverse solutions achieved from noise reduction (Fig. 4.12) are slightly better than those achieved from measurement spacing reduction (Fig. 4.10).

Finally, we test the inverse solutions obtained from improved measurement spacing and improved noise levels in sample schemes 7 and 8. The solutions illustrated in Figure 4.13, show no improvement over the solutions achieved by scheme 3 which used half the number of measurements with five times the level of noise. Again, this is because the center of the grain primarily holds information about the long-term integration of the time-temperature history. Sample scheme 8 however, which uses 24 measurements 6 microns apart taken at the grain boundary, results in very desirable time-temperature history reconstructions. As seen in Figure 4.15, the reheating event's temporal location, its peak temperature, and relative duration are all accurately recovered. Additionally, the linear cooling trend following the reheating event is almost exactly recovered. For inverse solutions this accurate though, the noise level needs to be reduced to ±0.06 ‰, which is not currently achievable using standards such as UWQ-1 Quartz which has an uncertainty of ±0.15‰.

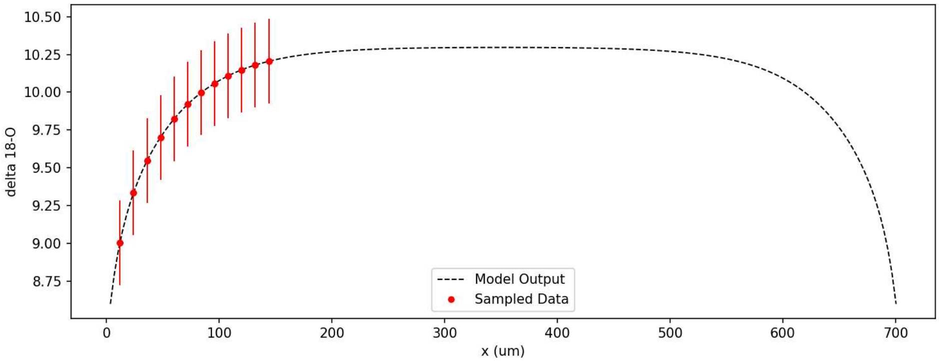
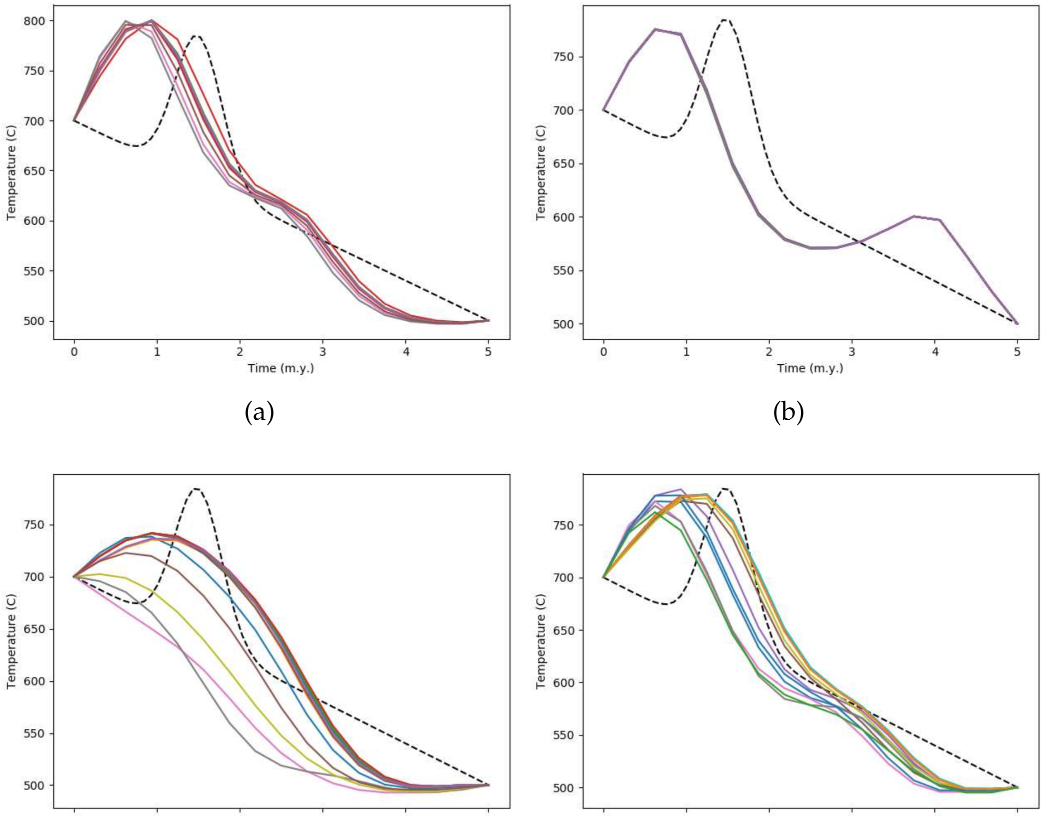


Figure 4.1: Measurement locations and uncertainties for sample scheme 1.



(a)

(b)

(c)

(d)

Figure 4.2: Four separate noise realizations for sample scheme 1.

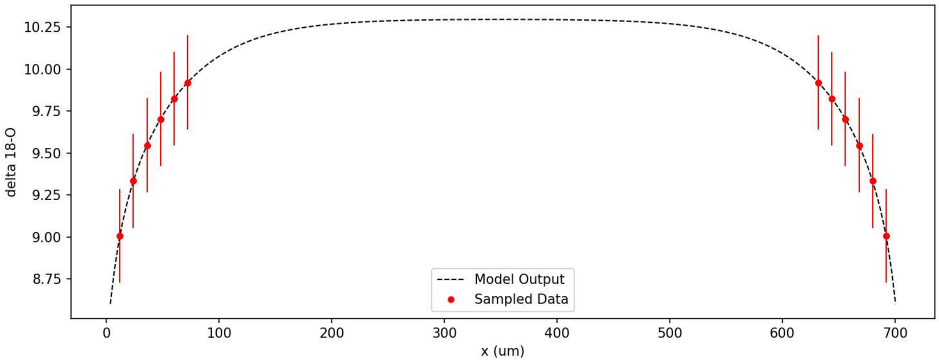
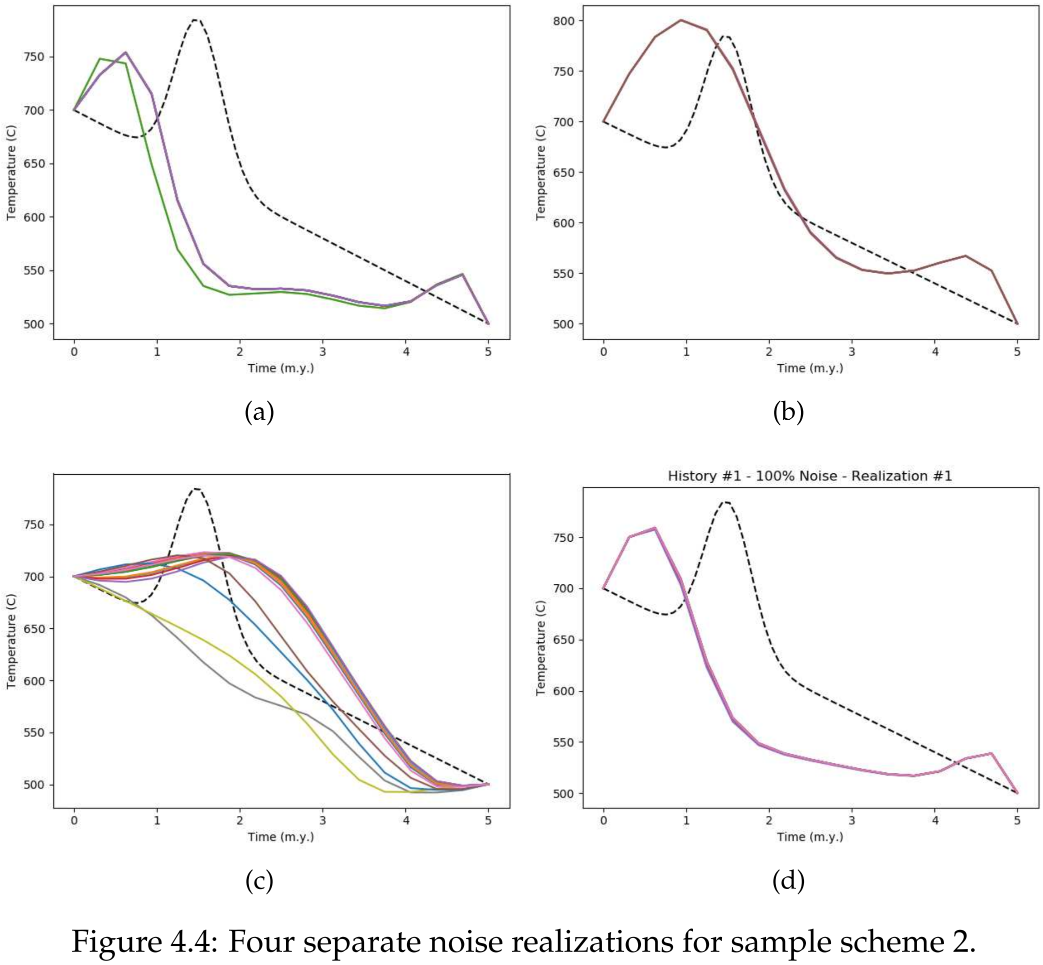


Figure 4.3: Measurement locations and uncertainties for sample scheme 2.



(a)

(b)

(c)

(d)

Figure 4.4: Four separate noise realizations for sample scheme 2.

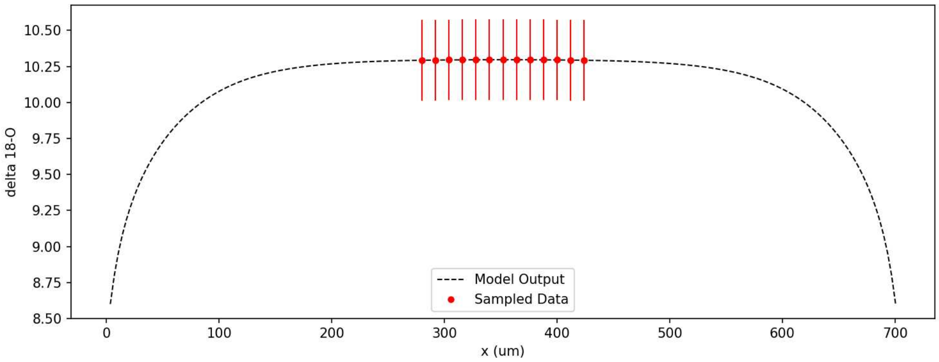
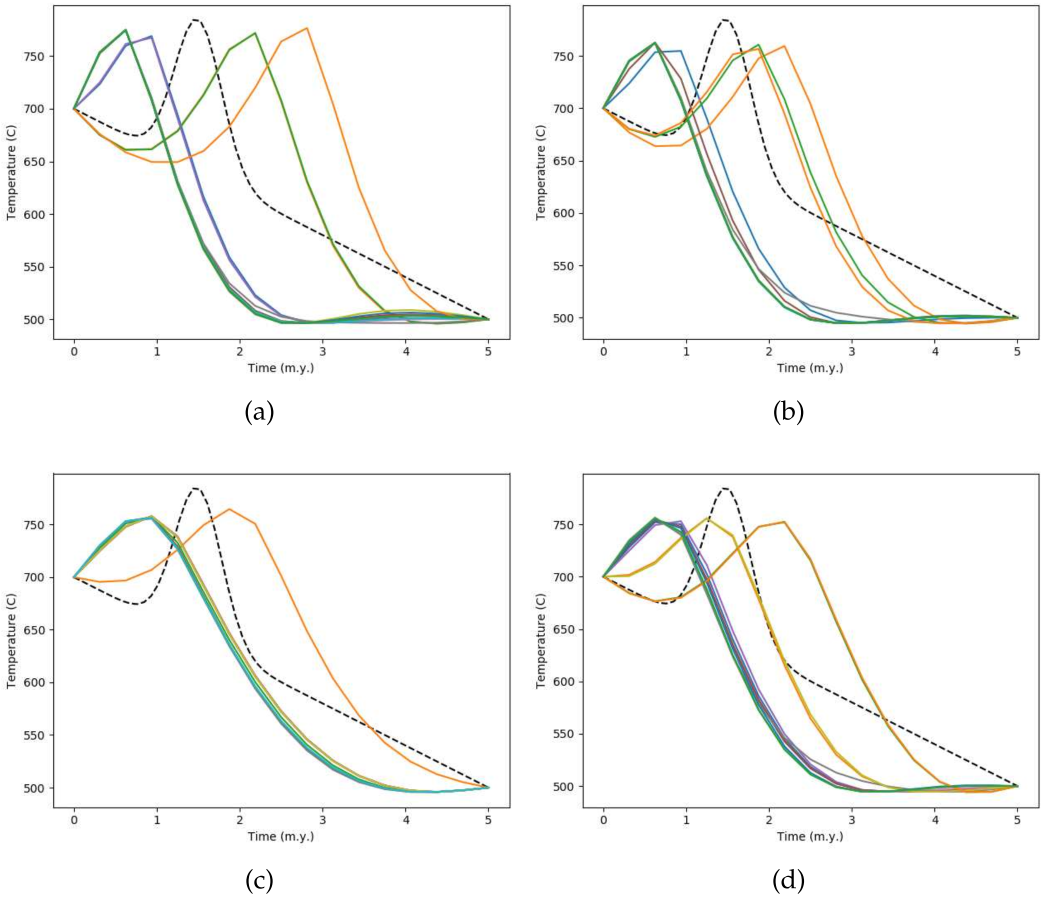


Figure 4.5: Measurement locations and uncertainties for sample scheme 3.



(a)

(b)

(c)

(d)

Figure 4.6: Four separate noise realizations for sample scheme 3.

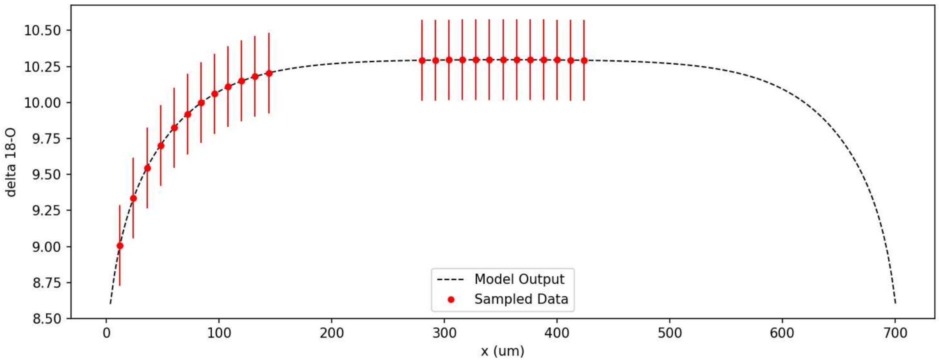
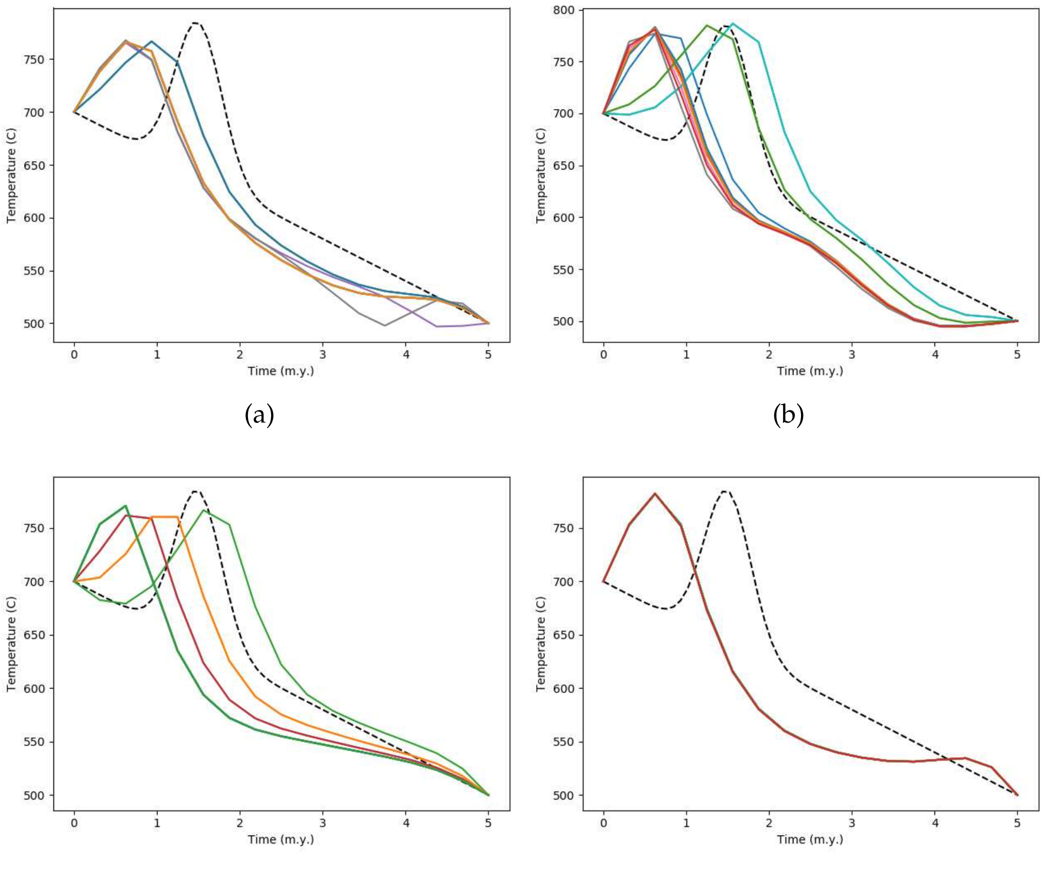


Figure 4.7: Measurement locations and uncertainties for sample scheme 4.



(a)

(b)

(c)

(d)

Figure 4.8: Four separate noise realizations for sample scheme 4.

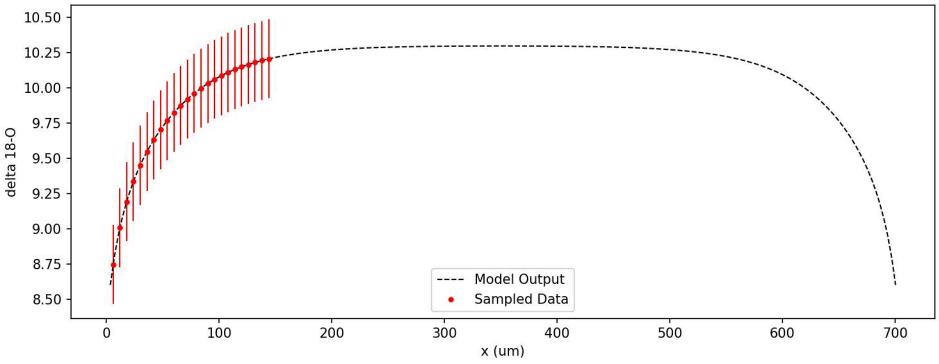
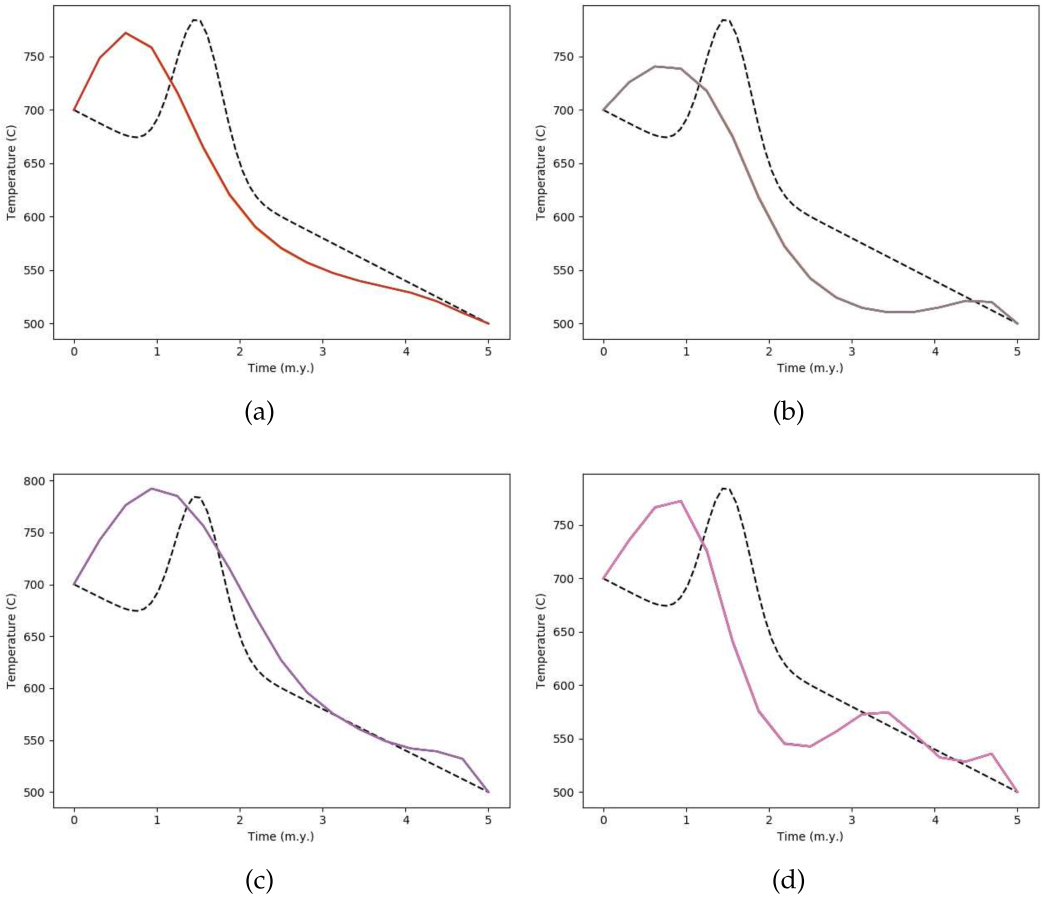


Figure 4.9: Measurement locations and uncertainties for sample scheme 5.



(a)

(b)

(c)

(d)

Figure 4.10: Four separate noise realizations for sample scheme 5.

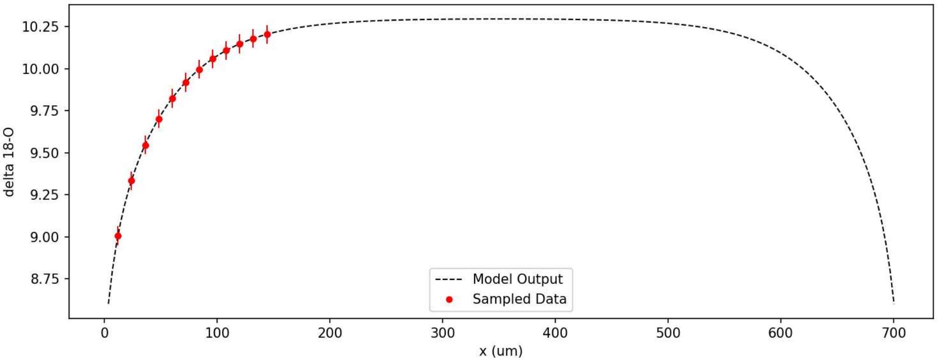
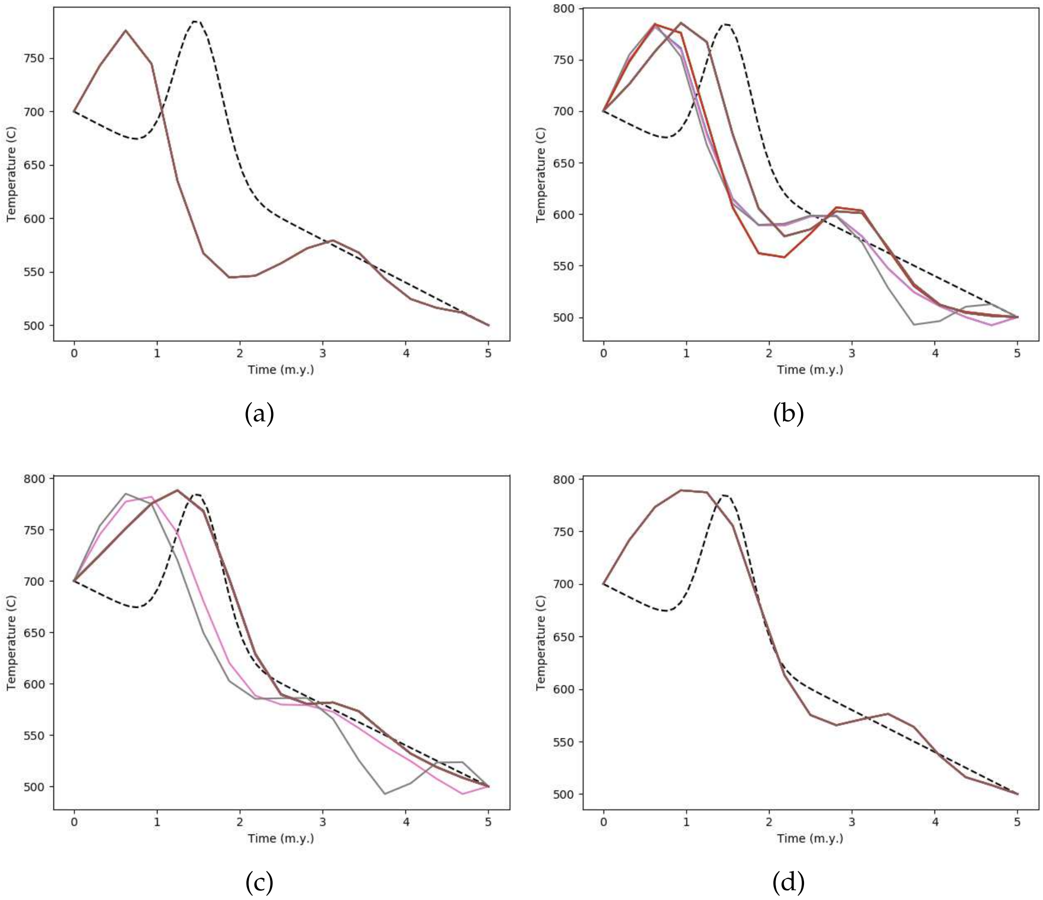


Figure 4.11: Measurement locations and uncertainties for sample scheme 6.



(a)

(b)

(c)

(d)

Figure 4.12: Four separate noise realizations for sample scheme 6.

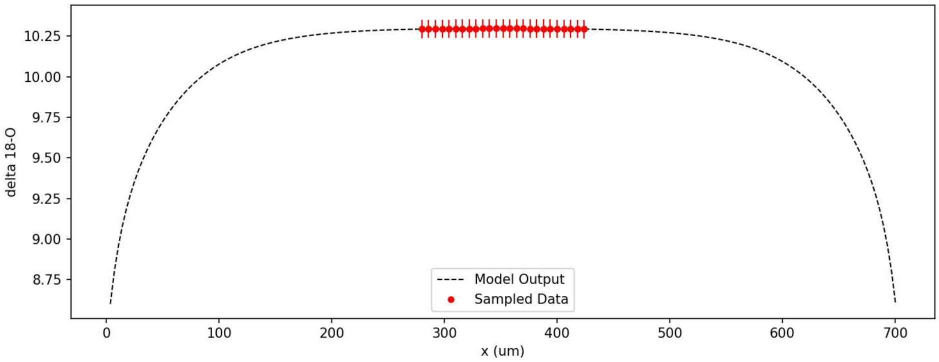
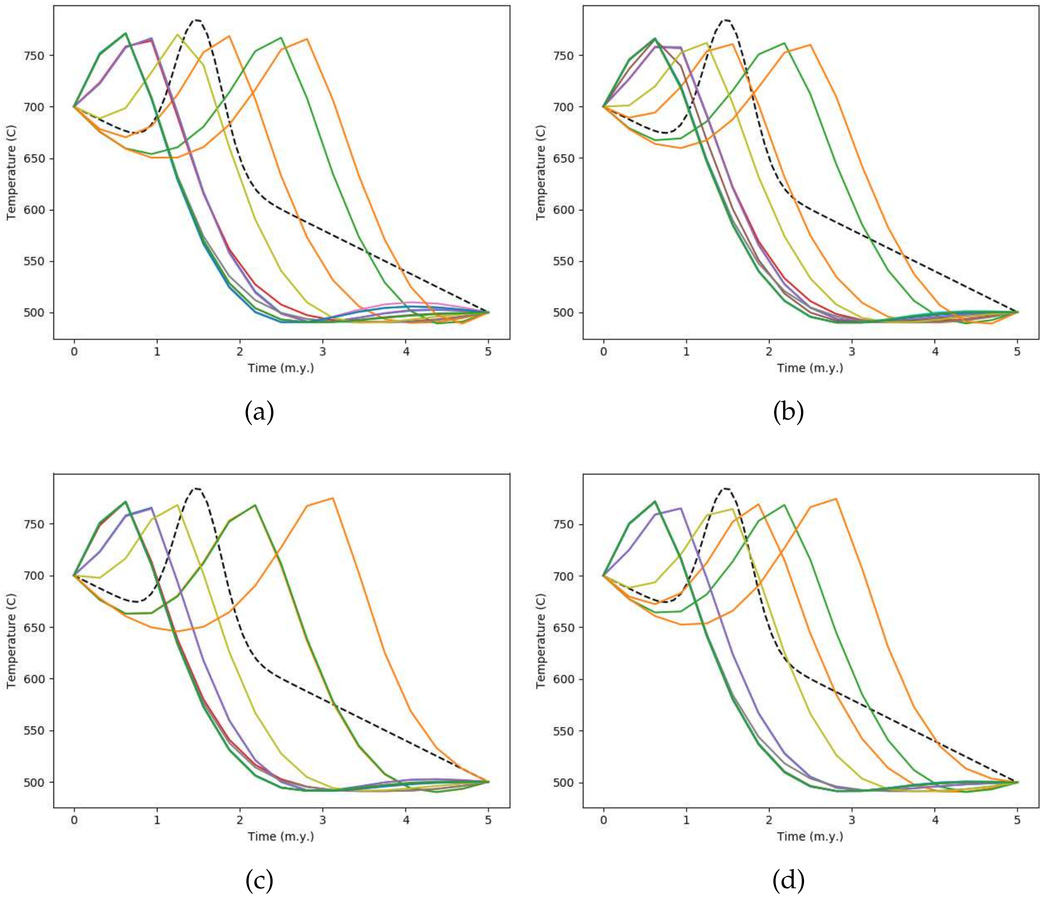


Figure 4.13: Measurement locations and uncertainties for sample scheme 7.



(a)

(b)

(c)

(d)

Figure 4.14: Four separate noise realizations for sample scheme 7.

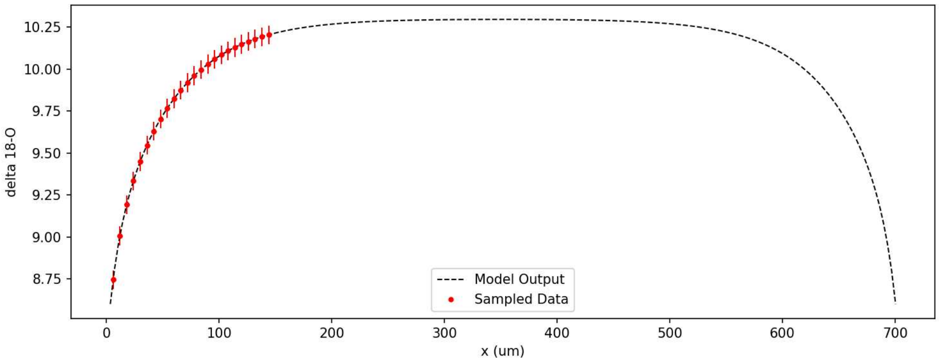
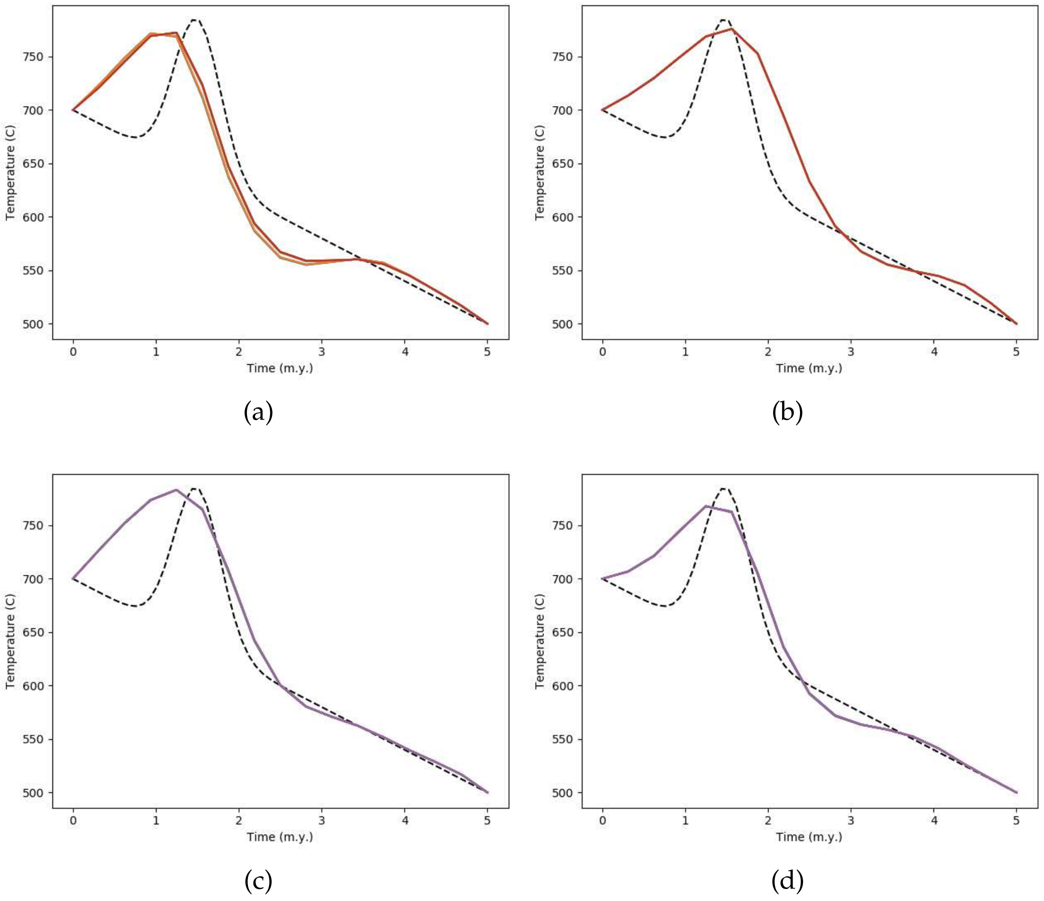


Figure 4.15: Measurement locations and uncertainties for sample scheme 8.



(a)

(b)

(c)

(d)

Figure 4.16: Four separate noise realizations for sample scheme 8.

CHAPTER 5

CONCLUSIONS

This thesis started by rebuilding the forward Fast Grain Boundary model of oxygen isotope exchange created by Eiler et al. (1994). The forward model was packaged into a distributable python repository which contains options for quick parameter referencing. Included in the repository is pre-compiled C code which can be used to accurately simulate 5 million years of diffusive isotope exchange in roughly 0.2 seconds. A GUI interface for the application was built so that all results demonstrated can be easily reproduced.

With the improved runtime achieved by our C code, we were able to implement the Levenberg-Marquardt optimization algorithm to recover time-temperature histories from diffusive isotope profiles. The inverse solver was built into the software application. Synthetic data was generated to test the inverse solver's capabilities at recovering a known thermal history. We showed that inverse solutions can be improved by measuring data closest to the mineral's grain boundary. We found that with the current level of noise and minimum distance between successive measurements we cannot reliably recover the location or duration of reheating events.

However, with sufficiently small enough noise in our oxygen isotope data, and implementing optimal sampling around the mineral's grain boundary, we could recover a reheating event's time of occurrence, its duration, and its peak temperature reliable through repeatable noise realizations. This required a noise level that is not currently achievable due to the uncertainties associated with the standards we measure against. Thus, if better standards are created and adopted, we can provide this type of accurate thermal history recovery in the future.

One weakness of the approach we have taken, is that the user must specify the timing of temperature values to recover and the total thermal history duration. We have not sufficiently explored how sensitive the solver is to histories that are longer or shorter than the durations we specify. One possible approach would be to include every temperature's associated time as an additional model parameter. However, this would dramatically increase the covariance between model parameters and double the computational complexity. Additionally, we did not optimize over the number of model parameters. This was chosen very qualitatively at the beginning research stages by choosing a number of model parameters that allowed adequate model complexity but still resulted in consistent convergence. As the number of model parameters was moved past 40 the inverse solver began returning nonsensical time-temperature histories as numerical rounding errors began to plague the iterative steps of the LM algorithm.

Finally, there is still a large body of work that needs to be done to validate the inverse solver's solutions; and improved standards are needed. However, this research shows that there is a significant amount of thermal information stored in the isotopic records, and with sufficient standards and measurement techniques, this information can be extracted to recover accurate continuous time-temperature histories.

APPENDIX A

PERMISSIONS

As quoted from direct email correspondence with the image's owner, Chloe Bonamici: "I grant you permission to use images that I have created and that appear in Figures 1.1, 1.2, and 2.1 of your M.S. thesis. I hold the sole copyright to these images, and they have not been published elsewhere."

All other figures and images were personally created for the sole purpose of appearing in this thesis document.

REFERENCES

Aster, R. C., Borchers, B., and Thurber, C. (2018). *Parameter estimation and inverse problems*. Third edition.

Dodson, M. H. (1973). Closure temperature in cooling geochronological and petrological systems.

Dodson, M. H. (1986). Closure profiles in cooling systems.

Ehlers, T. A., Chaudhri, T., and Kumar, S. (2005). Computational tools for low-temperature thermochronometer interpretation.

Eiler, J. M., Baumgartner, L. P., and Valley, J. (1994). Fast grain boundary: a fortran-77 program for calculating the effects of retrograde interdiffusion of stable isotopes.

Farley, K. (2000). Helium diffusion from apatite: general behavior as illustrated by durango fluorapatite.

Gallagher, K. (2011). Transdimensional inverse thermal history modeling for quantitative thermochronology.

Green, P., Duddy, I., Laslett, G., and Hegarty, K. (1989). Thermal annealing of fission tracks in apatite 4. quantitative modelling techniques and extension to geological timescales.

Hoefs, J. (2009). *Stable isotope geochemistry*. Springer, sixth edition.

Jager, E. and Niggli, E. (1964). Rubidiumstrontium-isotopenanalysen an mineralien und gesteinen des rotondogranites und ihre geologische interpretation.

Ketcham, R. A. (2005). Forward and inverse modeling of low-temperature thermochronometry data.

Lovera, O. M., Richter, F. M., and Harrison, T. M. (1989). The 40Ar/39Ar thermochronometry for slowly cooled samples having a distribution of diffusion domain sizes.

Shewmon, P. (2016). *Diffusion in solids*. Springer.

Strikwerda, J. C. (2004). *Finite difference schemes and partial differential equations*, volume 88. Siam, second edition.

Zeitler, P., Herczeg, A., McDougall, I., and Honda, M. (1987). U-Th-He dating of apatite: A potential thermochronometer.

THERMAL HISTORY INVERSION USING THE FAST GRAIN BOUNDARY MODEL OF DIFFUSION

by

Gabriel Kropf

NEW MEXICO TECH
SCIENCE • ENGINEERING • RESEARCH UNIVERSITY