



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL

A New Suite of Statistical Algorithms for Bayesian Model Fitting with Both Intrinsic and Extrinsic Uncertainties in Two Dimensions

Nicholas Clayton Konz

Senior Honors Thesis

submitted to

Department of Physics and Astronomy

The University of North Carolina at Chapel Hill

Approved for Highest Honors:

Prof. Daniel Reichart, Thesis Advisor

Prof. Fabian Heitsch, Reader/Committee Member

Prof. Lu-Chang Qin, Reader/Committee Member

Chapel Hill, April 2020

This document is set in Palatino, compiled with pdfL^AT_EX2e and Biber.

The L^AT_EX template from Karl Voit is based on KOMA script and can be found online:
<https://github.com/novoid/LaTeX-KOMA-template>

Honor Code

I declare that I have authored this thesis independently, that I have not used other than the declared sources/resources, and that I have explicitly indicated all material which has been quoted either literally or by content from the sources used.

Date

Signature

Abstract

Fitting a statistical model to data is one of the most important tools in any scientific or data-driven field, and rigorously fitting a two dimensional model to data that has intrinsic uncertainties (error bars) in both the independent variable and the dependent variable is a daunting task, especially if the data also has extrinsic uncertainty (sample variance) that cannot be fully accounted for by the error bars. Here, I introduce a novel statistic (described as the Trotter, Reichart, Konz statistic, or TRK) developed in Trotter (2011) that is advantageous towards model-fitting in this “worst-case data” scenario, especially when compared to other methods. I implemented this statistic as a suite of fitting algorithms in C++ that comes equipped with many capabilities, including: support for any nonlinear model; probability distribution generation, correlation removal and custom priors for model parameters; asymmetric uncertainties in the data and/or model, and more. I also built an end-to-end website through which the algorithm can be used easily, but generally, with a high degree of customizability. The statistic is applicable to practically any data-driven field, and I show a few examples of its usage within the realm of astronomy. This thesis along with Trotter (2011) form the foundations for Trotter, Daniel E. Reichart, and Konz (2020), in preparation. The TRK source code and web-based calculator can be found at <https://github.com/nickk124/TRK> and <https://skynet.unc.edu/rcr/calculator/trk>, respectively.

Acknowledgements

To begin, I would like to acknowledge the support of the excellent Department of Physics and Astronomy at UNC Chapel Hill, for its vibrant and closely-knit community of students and faculty, its many excellent and brilliant professors, and the wonderful atmosphere of learning and collaboration that is facilitated by everyone within. Without such a positive and encouraging context, my research would have never become what it is today. Similarly, I would also like to thank current and past members of the Reichart Lab/Skynet Robotic Telescope Network for their support and collaboration throughout my time in the lab, including Dylan Dutton, Vladimir Kouprianov, John Martin, and Roark Habegger. I would also like to specifically thank Prof. Fabian Heitsch, as both his course on physical modeling and my consulting of his vast knowledge of numerical techniques and modeling for this work equipped me to tackle a number of challenging problems. I would also like to acknowledge the financial support of the NASA/North Carolina Space Grant Undergraduate Research Fellowship and Scholarship Programs, as well as the UNC Department of Physics and Astronomy in the form of the Earl Nelson Mitchell Scholarship in Physics, and the Robert Shelton Award for Outstanding Research.

Without Adam Trotter, this thesis would not exist, and not just because his 2011 PhD dissertation serves as the foundation for this work. Ever since I began work on this project, Adam has been incredibly helpful and insightful. His attentiveness and rapid response time toward my constant volley of questions is only matched by his immediate dedication toward solving any problem I level at him. I very much look forward to continuing to work with Adam in the future.

Dr. Daniel Reichart. Prof. Reichart, or Dan, is a man of many names and of many interests. A theorist by training, he has worked with observational and radio astronomy, statistics,

astronomy teaching, and many other disciplines. I am continually amazed at the veritable font of knowledge that he possesses, and his willingness to dive deep into any problem, straight to the whiteboard, no matter how difficult it becomes. I have never met someone with such rapid, full attentiveness and masterful way of elucidating difficult topics and quickly figuring out (often unorthodox) solutions to new problems. Overall, Dan has not just been indispensable to helping me with this project; he has been the driving motivator for my entire research career over these past three years. Anybody would be extremely lucky to have an advisor half as good as him.

Contents

Abstract	iv
Acknowledgements	v
1. Introduction	1
2. The TRK Statistic	3
2.1. Statistical Preliminaries: Bayesian Statistics	3
2.2. Fitting a Model to Data in Two Dimensions	4
2.2.1. The Likelihood Function	6
2.2.2. An Analytical Approximation of the Likelihood	10
2.2.3. The TRK Likelihood	13
3. Properties of the TRK Statistic	16
3.0.1. Invertibility	16
3.0.2. Scalability	16
3.0.3. The TRK Correlation Coefficient	17
4. The TRK Codebase: Core Algorithms	21
4.1. Tangent Point Finding	22
4.2. Likelihood Maximization	24
4.3. Scale Optimization	29
4.4. Model Parameter Distribution and Uncertainty Computation	32
4.4.1. Using Adaptive MCMC to Sample Parameter Distributions	32
4.4.2. Computing Uncertainties from Parameter Distributions	36
4.4.3. Reality Check: A Linear TRK Example Fit	39

5. The TRK Codebase: Additional Algorithms	41
5.1. Automatic Model Parameter Correlation Removal and Pivot Points . . .	41
5.2. Asymmetric Intrinsic and/or Extrinsic Uncertainty	46
5.2.1. Theory	49
5.2.2. Implementation	55
6. Applications and Examples	56
6.1. Comparison to Similar Algorithms	56
6.2. Interstellar Extinction Model Parameter Fits	58
6.2.1. Fitting c_1 vs. c_2	61
6.2.2. Fitting BH vs. c_2	64
6.3. A Web-Based TRK Fit Calculator	65
7. Future Endeavors	73
7.1. A Scale Optimization Algorithm for Asymmetric Uncertainties	73
7.2. Support for N -dimensional Models	73
7.3. Python Implementation	74
A. Additional Algorithm Listings	76
Bibliography	80

List of Figures

2.1.	Example dataset and underlying model distribution where the scatter of the data cannot solely be accounted for by the error bars, which must be parameterized as extrinsic scatter, or <i>slop</i> . Model distribution is shown with 1–, 2– and 3σ confidence regions for the <i>slop</i> of the model, to properly account for the uncertainty in the dataset.	5
2.2.	Visualization of some two-dimensional dataset (from Trotter (2011)) with both intrinsic and extrinsic scatter in two dimensions, and accompanying model distribution with 1–, 2– and 3σ confidence regions for the extrinsic scatter/ <i>slop</i> of the model represented by the shaded regions. Inset box is shown zoomed in Fig. 2.3.	7
2.3.	Fig. 2.2, zoomed in, from Trotter (2011). Centered is a single datapoint (x_n, y_n) with intrinsic probability distribution defined by its' error bars/intrinsic scatter $\sigma_{x,n}, \sigma_{y,n}$, alongside a model distribution with curve $y_c(x; \vartheta_m)$ and extrinsic scatter/ <i>slop</i> parameters (σ_x, σ_y) . The shaded regions represent the 1–, 2– and 3σ confidence regions of the datapoint's intrinsic probability distribution, and of the extrinsic scatter-convolved model distribution.	8
2.4.	Visualization of some datapoint (x_n, y_n) with convolved error ellipse defined by the extrinsic-intrinsic convolved error bars $(\Sigma_{x,n}, \Sigma_{y,n})$, alongside some model curve $y_c(x)$. Also shown is the approximation of the non-linear model as a line tangent to the error ellipse at point $(x_{t,n}, y_{t,n})$	11

List of Figures

2.5. Illustration of the geometry of the TRK statistic modified from Trotter (2011), given some datapoint and model. The datapoint is centered at (x_n, y_n) (point O), with convolved error ellipse described by the widths $(\Sigma_{x,n}, \Sigma_{y,n})$ (Equation (2.9)). The model curve $y_c(x; \vartheta_m)$ is tangent to the convolved error ellipse at tangent point $(x_{t,n}, y_{t,n})$ (point T), and the red line is the linear approximation of the model curve, with slope $m_{t,n} = \tan \theta_{t,n}$. The blue line indicates the rotated coordinate axis u_n for the TRK statistic, perpendicular to the v_n axis.	15
4.1. Example of a model curve $y_c(x; \vartheta_m)$ (purple, top) and datapoint (x_n, y_n) with convolved error ellipse (red, middle) described by $(\Sigma_{x,n}, \Sigma_{y,n})$ where there are multiple points where y_c is tangent to the ellipse. This translates to there being multiple solutions/roots to/of Equation (2.13), which is plotted in orange at the bottom.	24
4.2. Example of slop/extrinsic scatter (σ_x, σ_y) dependence on fitting scale s (for some linear model). Note that as is predicted in §3.0.3, there exist minimum and maximum scales $s = a$ and $s = b$ (respectively) such that $\lim_{s \rightarrow a^+} \sigma_x = 0$ and $\lim_{s \rightarrow b^-} \sigma_y = 0$	30
4.3. Histograms of MCMC Metropolis-Hastings samplings for a normal-distributed parameter m , with true Gaussian posterior plotted in red, given (Gaussian) proposal distribution widths/“step sizes” of $\sigma_m = 10$, $\sigma_m = 0.1$, and $\sigma_m = 0.0001$, from left to right, given a sample size of $R = 100,000$. Note that even with such a high sample size, the choice of the step size for the model parameter’s Markov Chain will <i>drastically</i> change the quality of the sampling.	35
4.4. Top left: Example extrinsic scatter-dominated dataset (red) with small x - and y - error bars (barely visible), alongside linear fit model distribution $y = mx + b$ with 1-, 2- and 3σ slop confidence regions visible in blue. Top right: model parameter confidence ellipse with 1-, 2- and 3σ regions shaded, generated from the posterior probability distributions for b and m found via MCMC, bottom.	40

List of Figures

- 5.1. From Trotter (2011), this figure illustrates two different linear fits on the same dataset, with both sets of model parameters b, m offset from some best fit b, m , representing the uncertainty in this best fit. As shown, both of these lines intersect at some optimum pivot point x_p , plotted in red; if this optimal x_p is chosen when fitting the model, the correlation between b and m will be minimized. 42
- 5.2. Ten different iterations for the MCMC-generated distributions of new pivot points x_p^{new} computed with Equation (5.3) and weighted according to Equation (5.7), overlapped. For each iteration we take the optimal value of the new pivot point to be the weighted half-sample mode (see Bickel and Fruehwirth (2005)) of that iteration's distribution, and use this optimal value as x_p^{old} for the next iteration in order to compute the next distribution of x_p^{new} . As can be seen (given that only four or so iterations are visible), I have found this algorithm to converge quickly, usually only needing a few iterations, if more than one is needed at all. 45
- 5.3. Top left: The same extrinsic scatter-dominated dataset as in Figure 4.4 but now with pivot point x_p in model $y = m(x - x_p) + b$ optimized to minimize correlation between b and m , using Algorithm 8. Model is shown with 1-, 2- and 3σ slop confidence regions visible in blue. Top right: model parameter confidence ellipse with 1-, 2- and 3σ regions shaded, generated from the posterior probability distributions for b and m found via MCMC, bottom. Note that with pivot point optimized, the confidence ellipse is now no longer tilted as in Fig. 4.4, indicating the removal of correlation, also evidenced by a Pearson correlation coefficient of $R^2 = 0.056 \simeq 0$ 48

List of Figures

- 5.4. From Trotter (2011), a visualization of the 2D joint probability for an asymmetric model curve y_c and data-point (x_n, y_n) with asymmetric error bars. This joint probability is an integral that can be approximated by a 2D asymmetric Gaussian with widths $(\Sigma_{x,n\pm}, \Sigma_{y,n\pm})$ centered at some $(x_n + \delta_{x,n}, y_n + \delta_{y,n})$. The integral is broken into three segments according to the quadrants about the convolved centroid $(x_n + \delta_{x,n}, y_n + \delta_{y,n})$, in this case I_{-+} , I_{++} and I_{+-} , respectively, with subscripts denoting which quadrant the integral/linear approximation of the model curve is located in. For this example, I_{-+} (red) through quadrant 2 has limits $(-\infty, x_1]$, I_{++} (green) through quadrant 1 has limits $[x_1, x_2]$, and I_{+-} (blue) through quadrant 4 has limits $[x_2, \infty)$ 50
- 6.1. From Trotter (2011), the combined CCM and FM extinction model curve along a typical Milky Way line of sight. The y -axis is defined to be the ratio between the $\lambda - V$ color excess $E(\lambda - V)$ at a given wavelength λ , and the $B - V$ color excess $E(B - V)$, where the color excess is defined as the difference, in magnitudes, of the absorption due to dust at two given wavelengths. In this case, B and V correspond to wavelengths in the middle of standard blue and visible photometric filters. The color excess ratio plotted on the y -axis is thus proportional (with an offset) to the magnitude of dust extinction at a given wavelength λ . The x -axis is proportional to the inverse of λ , specifically $x \equiv (\lambda/1 \mu\text{m})^{-1}$, or, equivalently, proportional to frequency. As shown, the parameters c_1 and c_2 describe the intercept and slope of a linear component of the model. The parameter γ parameterizes the width of the “bump” in the center of the model (also known as the “UV bump”), while the parameter $\text{BH} \equiv c_3/\gamma^2$ describes the height of this bump. 60
- 6.2. Observed c_1 vs c_2 data from Valencic, Clayton, and Gordon (2004), Gordon et al. (2003), E. Fitzpatrick and Massa (2007), and Clayton et al. (2015), plotted with linear TRK fit modeled by Equation (6.4). Shaded regions indicate the 1–, 2– and 3σ slop confidence regions of the model distribution, given best fit slop values of Table 6.2.1 and plotted according to footnote 19 on page 39. 62

List of Figures

6.3.	MCMC-generated (§4.4.1) probability distributions for c_1 vs. c_2 (Equation (6.4)) model (top) and slop/extrinsic scatter (bottom) parameters. Parameter confidence ellipses (center) with 1–, 2– and 3σ regions show joint posterior probabilities with respect to the parameters plotted on either side. The pivot-point finding algorithm of §5.1 was used to remove correlation between model parameters.	63
6.4.	Observed BH vs c_2 data from Valencic, Clayton, and Gordon (2004), Gordon et al. (2003), E. Fitzpatrick and Massa (2007), and Clayton et al. (2015), plotted with broken-linear TRK fit modeled by Equation (6.5). Shaded regions indicate the 1–, 2– and 3σ slop confidence regions of the model distribution, given best fit slop values of Table 6.2.2 and plotted according to footnote 19 on page 39.	66
6.5.	MCMC-generated (§4.4.1) probability distributions for BH vs. c_2 (Equation (6.5)) model (top and middle rows) and slop/extrinsic scatter (bottom row) parameters. Parameter confidence ellipses (center) with 1–, 2– and 3σ regions show joint posterior probabilities with respect to the parameters plotted on either side. The pivot-point finding algorithm of §5.1 was used to remove respective correlations between model parameters of each linear "leg" of the model curve.	67
6.6.	The data input step of the web-based TRK calculator with example data.	68
6.7.	Example of plotting input data on the TRK calculator webpage.	69
6.8.	Section of the TRK webpage where the user can choose which model to fit to their data.	70
6.9.	Section of the TRK webpage where the user can determine which additional TRK algorithms to run alongside basic fit.	71
6.10.	Output section of the TRK calculator webpage, showing the results of an example linear fit (without model parameter uncertainty computation). .	72

1. Introduction

Robustly fitting a statistical model to data is a task ubiquitous to practically all data-driven fields, but the more nonlinear, uncertain and/or scattered the dataset is, the more difficult this task becomes. In the common case of two dimensional models (i.e. one independent variable x and one dependent variable $y(x)$), datasets with intrinsic uncertainties, or error bars, along both x and y prove difficult to fit to in general, and if the dataset has some *extrinsic* uncertainty/scatter (i.e., sample variance) that cannot be accounted for solely by the error bars, the difficulty increases still.

In this work, I describe a new, easily generalizable statistic developed by Trotter (2011), the Trotter-Reichart-Konz statistic—hereafter the TRK statistic—that is used to fit models to data in such “worst case” scenarios. Such model *distributions* are defined by convolving a model curve $y_c(x; \vartheta_m)$ that describes the shape of the model with a 2D probability distribution that characterizes the scatter of the data, where ϑ_m is the set of parameters describing the model. In order to fit such a model to a set of data, values for ϑ_m and the parameters that describe the extrinsic scatter of the dataset must be determined that maximize the joint probability, or likelihood, of the model curve and the dataset; in other words, a model distribution must be found that is *most likely* to reproduce the dataset. Assuming that the intrinsic and extrinsic uncertainties are Gaussian (possibly asymmetric), I show how to define such a likelihood function following Trotter (2011). This likelihood includes integrating over a rotated coordinate system of choice, and when a specific set of coordinates is used, it results in a statistic that is invertible (i.e. fitting x vs. y will result in the same fit as y vs x), computationally feasible, and reduces to a χ^2 statistic in the classic 1D uncertainty case; we define this as the TRK statistic. Models predicted by maximizing the TRK statistic’s likelihood function are geometrically equivalent to models that minimize the sum of the squares of the radial distances of each

1. Introduction

datapoint centroid from the model curve, which I demonstrate. As such, the statistic is χ^2 -like, measured in the direction of closest approach of the model curve to the datapoint. The TRK statistic is not *scalable*, i.e. it yields different best fits depending on the choice of basis, or *scale* for each coordinate axis, with some optimum scale corresponding to the true best fit. However, I present an implementation of an algorithm originally conceptualized in Trotter (2011) that can determine such an optimum scale, effectively removing this caveat of the TRK statistic.

The original introduction of the TRK statistic in Trotter (2011) used a genetic algorithm-based implementation of the statistic that was non-automated, non-“production style” code that only demonstrated the proof-of-concept of the statistic, rather than introduced an easy-to-use, widely applicable codebase. Because of this, my contribution to TRK was to implement the statistic from scratch into a fully-fledged, general nonlinear fitting suite that can be picked up and used easily by anyone, while supporting a number of features. This codebase is the focus of this work, and can be found, with documentation, at <https://skynet.unc.edu/rcr/calculator/downloads>.

In §2 and §3 I introduce the TRK statistic and its central properties, closely following Trotter (2011). Following this, in §4 I introduce the core algorithms used to perform fits and other procedures with the TRK statistic in practice, including, but not limited to, fit scale optimization, determining best fits, and model parameter distribution generation. Then, in §5 I introduce additional algorithms of the TRK suite, including methods for the removal of correlation between model parameters, and the addition of asymmetric extrinsic and/or intrinsic uncertainties.

In §6 I begin by comparing the TRK suite to similar algorithms (§6.1), and then present TRK fits used to model relationships between empirical parameters that describe the extinction of light by dust in the Milky Way as examples of the usage of the suite (§6.2). Next, I present a robust but easy-to-use implementation of the algorithm into a webpage-based calculator that I developed end-to-end (§6.3), which can be used for quick, reliable fitting while also possessing a number of features. Finally, in §7 I discuss potential expansions of the suite that may be explored in later works.

2. The TRK Statistic

2.1. Statistical Preliminaries: Bayesian Statistics

I begin by considering some set of observed data D and a set of parameters H that describes some hypothetical model for this data. From here, I define $p(H)$ to be the *prior* probability distribution function, or prior for short; this defines how any prior information known about the model parameters (before data collection) affects and/or constrains the value of such parameters. For example, if some parameter has a best fit value and uncertainty from a previous result, this can form the basis of a prior for that parameter. Next, we define the *likelihood* probability distribution function $\mathcal{L}(D|H)$, or likelihood for short, as the conditional probability of obtaining the observed data D given some model described by H ; this is the term that describes *how likely the model is to have generated the data*. Finally, in order to examine how some model described by H can arise from the data, we define the *posterior* probability distribution function $p(H|D)$, or posterior for short, to be the probability of obtaining certain model parameters H given the data D .

These quantities can all be related with Bayes' Theorem

$$p(H|D) = \frac{\mathcal{L}(D|H)p(H)}{p(D)}, \quad (2.1)$$

of which the proof is outside the scope of this work, but is found in any introductory probability course. In Equation (2.1) the normalization factor $p(D)$, known as the *evidence*, is typically difficult to compute. However, really only being a normalization factor, it can safely be ignored for our fitting purposes (as will be shown in §4.2 and §4.4.1), such that

2. The TRK Statistic

in this work, Bayes' Theorem will be used in the form of

$$p(H|D) \propto \mathcal{L}(D|H)p(H). \quad (2.2)$$

From here, it's clear that given some prior(s) $p(H)$, if a likelihood function $\mathcal{L}(D|H)$ can be defined, then the posterior can be determined, which ultimately describes the distribution of the model parameters that arise from the dataset. The central goal of the remainder of this chapter will be to clearly define this likelihood.

2.2. Fitting a Model to Data in Two Dimensions

Consider a set of N two-dimensional datapoints $\{x_n, y_n\}$, where $n = 1, 2, \dots, N$. In the most general case, each datapoint has an intrinsic probability distribution defined by its *intrinsic scatter/statistical uncertainty*, or *error bars* for short, $\{\sigma_{x,n}, \sigma_{y,n}\}$ along one or both dimensions of x and y ¹. The dataset can also have *extrinsic scatter/sample variance*, or *slop* for short, again in both dimensions, that cannot solely be accounted for by the error bars (see Fig. 2.1 for an example). In this case, the slop must be parameterized and fit to as part of the model. Hereafter, unless otherwise stated, I assume that the slop is normally distributed along the two dimensions and described by the two parameters σ_x and σ_y , that are constant along the entire dataset. I also assume that the intrinsic scatter in both dimensions, and the extrinsic scatter in both dimensions, are respectively uncorrelated.

In the following section, we will begin to explore how a likelihood function describing a model for this “worst case” type of dataset can be defined, by connecting the extrinsic and intrinsic probability distributions of the dataset with the model itself.

¹The data can also be given weights $\{w_n\}$, i.e. assigning some datapoint a weight of 2 is equivalent to including that datapoint twice in the dataset.

2. The TRK Statistic

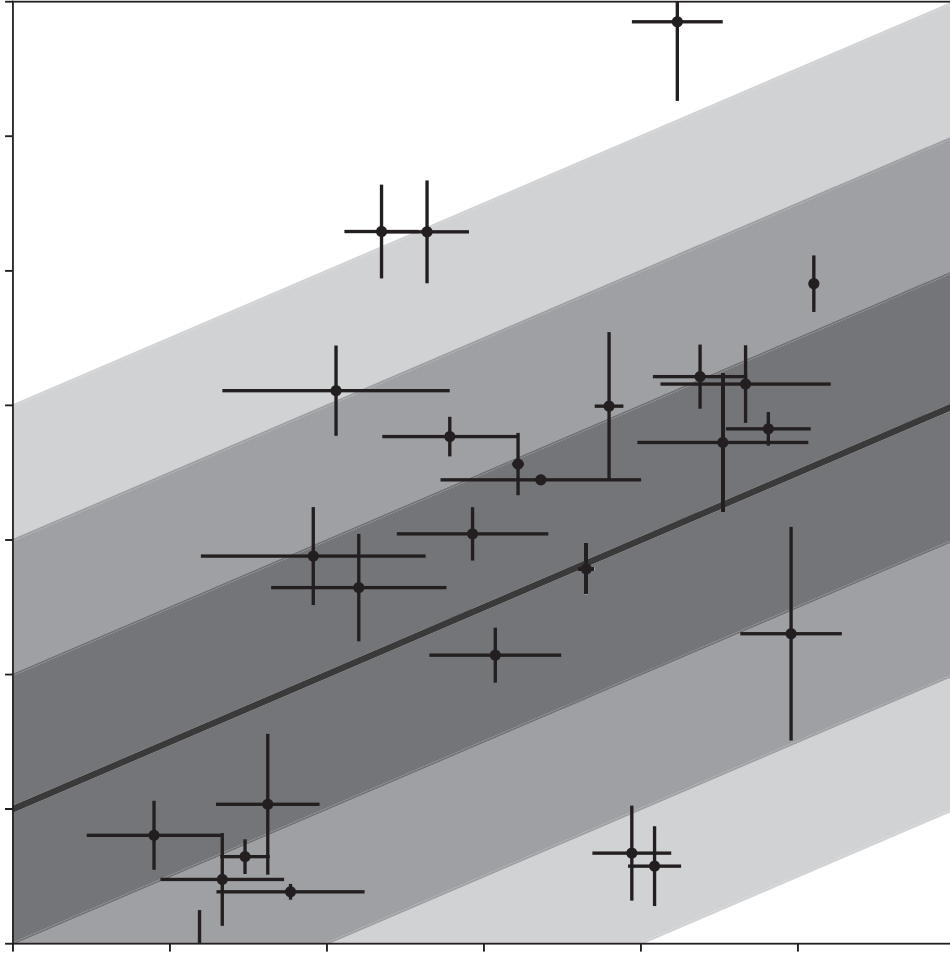


Figure 2.1.: Example dataset and underlying model distribution where the scatter of the data cannot solely be accounted for by the error bars, which must be parameterized as extrinsic scatter, or *slop*. Model distribution is shown with 1-, 2- and 3σ confidence regions for the *slop* of the model, to properly account for the uncertainty in the dataset.

2. The TRK Statistic

2.2.1. The Likelihood Function

The derivation below closely follows Trotter (2011), where the TRK statistic was initially formulated. In order to consider fitting a model to data, we must first determine how to quantify the goodness of fit for such a model, given some distribution of N measurements described in the previous section. I define part of the model as some probability distribution $g(x, y)$ that is convolved along some model curve function $y_c(x; \vartheta_m)$, where ϑ_m is the set of parameters that define the functional form of the model; here, $g(x, y)$ can be thought of as the "density function" of the model distribution. Then, in order to properly represent the scatter of the data, I define the full model distribution by convolving this with a 2D Gaussian distribution that characterizes the slop, with widths defined by the parameters σ_x and σ_y . This representation of the model can be difficult to conceptualize, but it is necessary in order to work with the most general, Bayesian treatment of a two-dimensional uncertain dataset. A visualization of this is shown in Figures 2.2 and 2.3.

Another way to express the (effectively 1D) model curve $y_c(x; \vartheta_m)$ is to describe it as a one-dimensional delta function along some arbitrary chosen (orthogonal) coordinate system (u_n, v_n) . Indicated by the subscripts of n , such coordinates can, in the most general case, vary between datapoints. As such, we can express the probability density along the model curve as $g(x, y)\delta(v_n, v_{c,n}(u_n; \vartheta_m))$, where $v_{c,n}(u_n; \vartheta_m)$ is $y_c(x; \vartheta_m)$ in the (u_n, v_n) coordinate system, and δ denotes the one-dimensional Dirac delta function. Also note that given that the coordinates (u_n, v_n) are assumed to be orthogonal, we can always express them as a rotation from (x, y) .

Now that we have an expression for the probability density along the model curve, we can convolve this with the extrinsic scatter/slop distribution (again, assumed to be 2D Gaussian) to obtain the *model distribution function* for a single datapoint,

$$p_n^{\text{mod}}(x', y' | \vartheta_m, \sigma_x, \sigma_y) = \int_{u_n} \int_{v_n} g(x, y) \delta(v_n, v_{c,n}(u_n; \vartheta_m)) \mathcal{N}(x' | x, \sigma_x) \mathcal{N}(y' | y, \sigma_y) dv_n du_n, \quad (2.3)$$

given the definition of the convolution of two bivariate functions, where the integrals are

2. The TRK Statistic

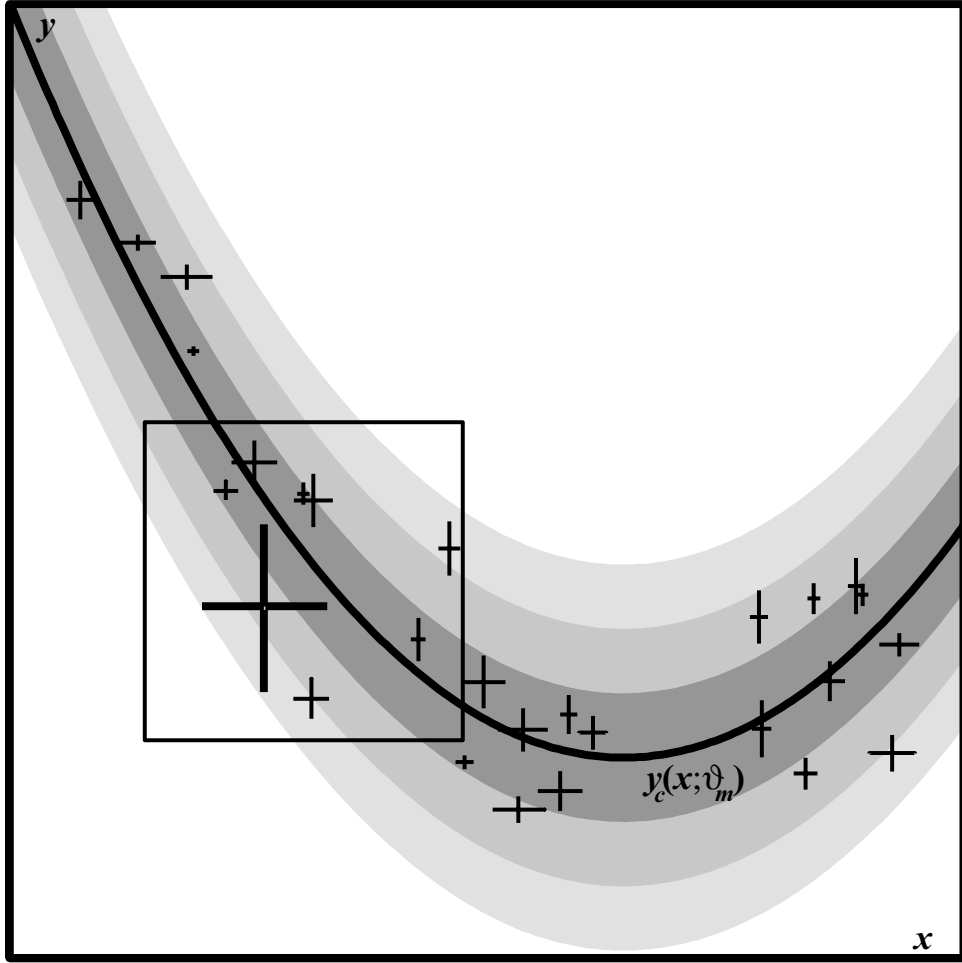


Figure 2.2.: Visualization of some two-dimensional dataset (from Trotter (2011)) with both intrinsic and extrinsic scatter in two dimensions, and accompanying model distribution with 1-, 2- and 3σ confidence regions for the extrinsic scatter/slop of the model represented by the shaded regions. Inset box is shown zoomed in Fig. 2.3.

2. The TRK Statistic

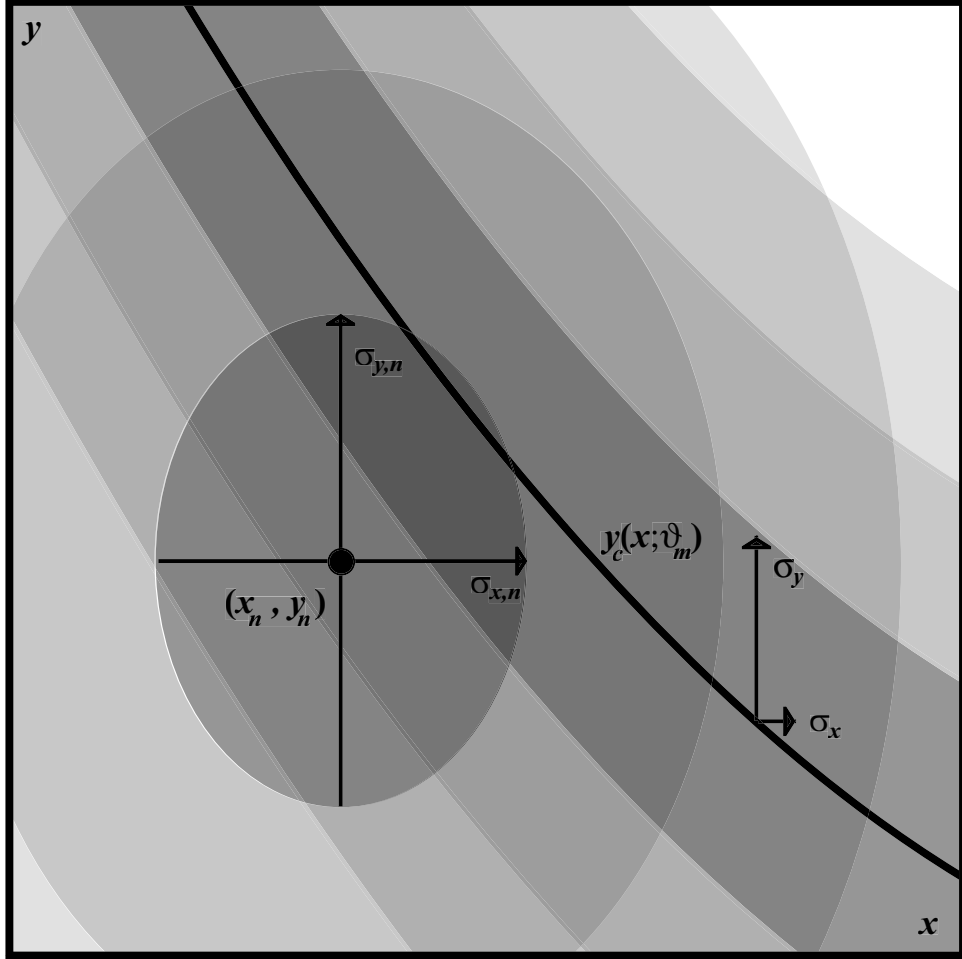


Figure 2.3.: Fig. 2.2, zoomed in, from Trotter (2011). Centered is a single data-point (x_n, y_n) with intrinsic probability distribution defined by its' error bars/intrinsic scatter $\sigma_{x,n}, \sigma_{y,n}$, alongside a model distribution with curve $y_c(x; \vartheta_m)$ and extrinsic scatter/slop parameters (σ_x, σ_y) . The shaded regions represent the 1-, 2- and 3σ confidence regions of the datapoint's intrinsic probability distribution, and of the extrinsic scatter-convolved model distribution.

2. The TRK Statistic

both over $(-\infty, \infty)$ and \mathcal{N} denotes the Gaussian/normal distribution

$$\mathcal{N}(x'|\mu, \sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2} \frac{(x' - \mu)^2}{\sigma^2}\right). \quad (2.4)$$

with mean μ and standard deviation σ .

Next, we need to obtain an expression for the full *joint* probability of a single datapoint with the model distribution function p_n^{mod} for that datapoint. The *intrinsic* probability for a single datapoint comes from the error bars for that datapoint, and is found with

$$p_n^{\text{int}}(x', y'|x_n, y_n, \sigma_{x,n}, \sigma_{y,n}) = \mathcal{N}(x'|x_n, \sigma_{x,n})\mathcal{N}(y'|y_n, \sigma_{y,n}). \quad (2.5)$$

again assuming Gaussian error bars. From here, we can find the joint probability of some n^{th} datapoint with the model distribution by integrating the product of the two distributions p_n^{mod} and p_n^{int} over x' and y' , as

$$p_n(\vartheta_m, \sigma_x, \sigma_y|x_n, y_n, \sigma_{x,n}, \sigma_{y,n}) = \int_{x'} \int_{y'} \int_{u_n} \int_{v_n} g(x, y) \delta(v_n - v_{c,n}(u_n; \vartheta_m)) \times \\ \mathcal{N}(x'|x, \sigma_x) \mathcal{N}(y'|y, \sigma_y) \mathcal{N}(x'|x_n, \sigma_{x,n}) \mathcal{N}(y'|y_n, \sigma_{y,n}) dv_n du_n dy' dx'. \quad (2.6)$$

Next, the likelihood function is defined to be the product of all N of the joint probabilities of the datapoints, as

$$\mathcal{L} = \prod_{n=1}^N p_n(\vartheta_m, \sigma_x, \sigma_y|x_n, y_n, \sigma_{x,n}, \sigma_{y,n}), \quad (2.7)$$

so in theory, our work of finding an expression for the likelihood is done. However, given the four integrals in Equation (2.6), this solution is quite computationally intractable. In order to obtain a practical likelihood, a few simplifying, but reasonable approximations need to be made, following Trotter (2011).

2.2.2. An Analytical Approximation of the Likelihood

To begin, we can simplify the expression for the joint probability p_n by noting that the (x', y') integral in Equation (2.6) can be evaluated analytically, which gives

$$p_n(\vartheta_m, \sigma_x, \sigma_y | x_n, y_n, \sigma_{x,n}, \sigma_{y,n}) = \int_{u_n} \int_{v_n} g(x, y) \delta(v_n - v_{c,n}(u_n; \vartheta_m)) \mathcal{N}(x | x_n, \Sigma_{x,n}) \mathcal{N}(y | y_n, \Sigma_{y,n}) dv_n du_n, \quad (2.8)$$

where $(\Sigma_{x,n}, \Sigma_{y,n})$ are the quadrature sums of both the intrinsic and extrinsic scatters:

$$\begin{aligned} \Sigma_{x,n} &\equiv (\sigma_{x,n}^2 + \sigma_x^2)^{1/2} \\ \Sigma_{y,n} &\equiv (\sigma_{y,n}^2 + \sigma_y^2)^{1/2}. \end{aligned} \quad (2.9)$$

What is the significance of these terms? In the words of Trotter (2011), Equation (2.8) indicates that the joint probability of some n^{th} datapoint with the model distribution is proportional to the integral of the effectively one-dimensional probability density along the model curve through a two dimensional *convolved* Gaussian, whose widths are the quadrature sums of the intrinsic and extrinsic uncertainties in each direction, $\Sigma_{x,n}$ and $\Sigma_{y,n}$. In other words, Fig. 2.3 is equivalent to 2.4.

To further simplify Equation (2.8) (i.e., remove the integrals), I'll begin by making the first approximation: that the intrinsic probability density along the model curve $g(x, y)$ varies slowly with respect to the scale of the size of the convolved error ellipse described by $\Sigma_{x,n}$ and $\Sigma_{y,n}$. This will make $g(x, y)$ approximately constant, such that it can be pulled out of the integral in Equation (2.8). Next, I will assume that the model curve $y_c(x; \vartheta_m)$ is approximately linear over this same scale; specifically, I will approximate y_c as a line $y_{t,n}(x)$ passing through the point $(x_{t,n}, y_{t,n})$ where y_c is tangent to the convolved error ellipse, with some slope $m_{t,n}$ (see Fig. 2.4), i.e.

$$y_c(x) \approx y_{t,n} + m_{t,n}(x - x_{t,n}). \quad (2.10)$$

If, at some position x , the model curve $y_c(x; \vartheta_m)$ is tangent to the error ellipse at some

2. The TRK Statistic

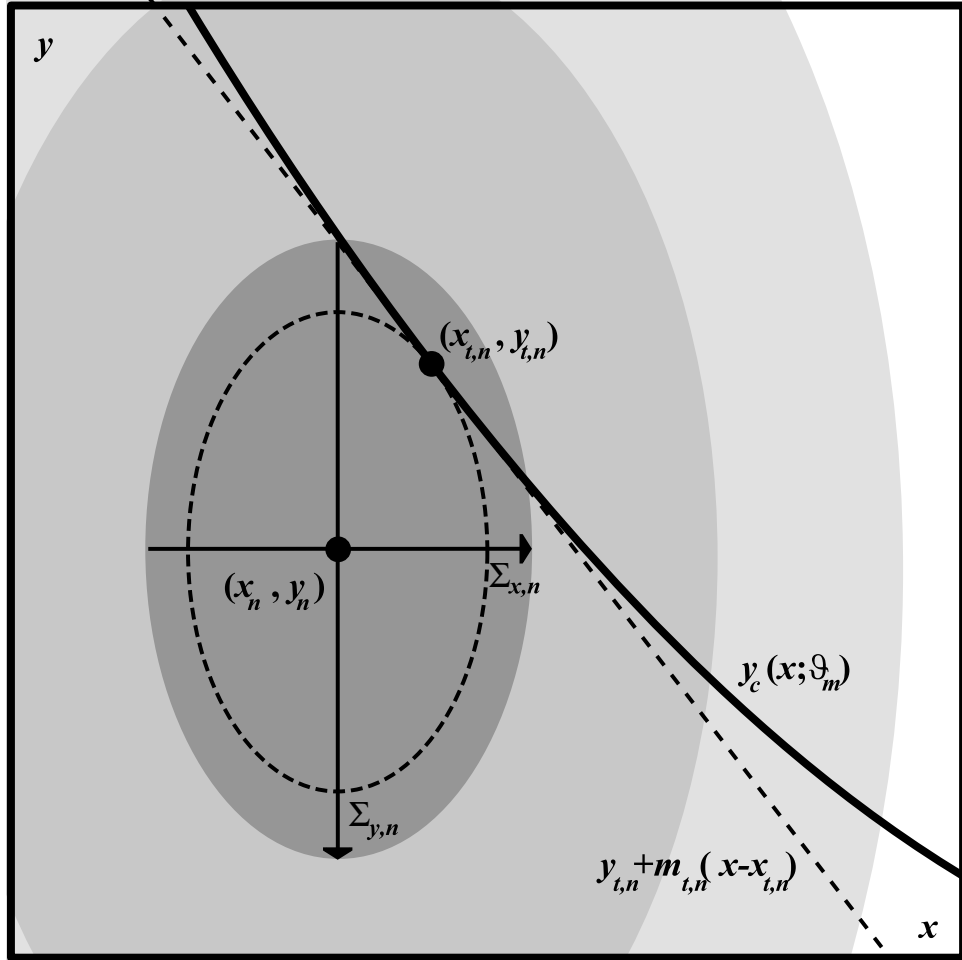


Figure 2.4.: Visualization of some datapoint (x_n, y_n) with convolved error ellipse defined by the extrinsic-intrinsic convolved error bars $(\Sigma_{x,n}, \Sigma_{y,n})$, alongside some model curve $y_c(x)$. Also shown is the approximation of the non-linear model as a line tangent to the error ellipse at point $(x_{t,n}, y_{t,n})$.

2. The TRK Statistic

$(x_{t,n}, y_{t,n})$, then we have the relation

$$\frac{(y_c(x; \vartheta_m) - y_n)^2}{\Sigma_{y,n}^2} + \frac{(x - x_n)^2}{\Sigma_{x,n}^2} = \frac{(y_{t,n} - y_n)^2}{\Sigma_{y,n}^2} + \frac{(x_{t,n} - x_n)^2}{\Sigma_{x,n}^2}, \quad (2.11)$$

which is the condition for $(x, y_c(x; \vartheta_m))$ to be a tangent point. Differentiating this equation with respect to x gives

$$\frac{d}{dx} \left(\frac{(y_c(x; \vartheta_m) - y_n)^2}{\Sigma_{y,n}^2} + \frac{(x - x_n)^2}{\Sigma_{x,n}^2} \right) = 0, \quad (2.12)$$

which implies that the tangent point is equivalent to the point on y_c that *minimizes the radial distance* to the centroid of the error ellipse. Evaluating the derivative gives us an equation that can be *implicitly* solved for $x = x_{t,n}$,

$$(y_c(x) - y_n) \frac{dy_c(x; \vartheta_m)}{dx} \Sigma_{x,n}^2 + (x - x_n) \Sigma_{y,n}^2 = 0, \quad (2.13)$$

given some model curve and datapoint.

Finally, given these two assumptions, the joint probability of the n^{th} datapoint and the model distribution of Equation (2.8) can be simplified by integrating over v_n , as

$$p_n(\vartheta_m, \sigma_x, \sigma_y | x_n, y_n, \sigma_{x,n}, \sigma_{y,n}) \approx g(x_n, y_n) \int_{-\infty}^{\infty} \mathcal{N}(x | x_n, \Sigma_{x,n}) \mathcal{N}(y_c(x; \vartheta_m) | y_n, \Sigma_{y,n}) du_n, \quad (2.14)$$

given the “selecting” property of the delta function on the Gaussian along y . By using the chain rule substitution $du_n = \frac{du_n}{dx} dx$, $y_c(x; \vartheta_m) \approx y_{t,n} + m_{t,n}(x - x_{t,n})$ from Equation (2.10), and integrating over x , we have finally arrived at an analytic expression for p_n .²

$$\begin{aligned} p_n(\vartheta_m, \sigma_x, \sigma_y | x_n, y_n, \sigma_{x,n}, \sigma_{y,n}) \\ \approx g(x_n, y_n) \frac{du_n}{dx} \mathcal{N} \left(y_n \mid y_{t,n} + m_{t,n}(x_n - x_{t,n}), \sqrt{m_{t,n}^2 \Sigma_{x,n}^2 + \Sigma_{y,n}^2} \right). \end{aligned} \quad (2.15)$$

The likelihood function is the joint probability of the model distribution with all of the

²Here, we have also implicitly made an additional approximation: that the efficiency of which the measured data samples the *true* model distribution is approximately constant along the scale(s) of $(\sigma_{x,n}, \sigma_{y,n})$ and (σ_x, σ_y) . This is unnecessary to delve into for the purposes of this work, but for an explicit inclusion of this, see §2.2.1 of Trotter (2011).

2. The TRK Statistic

datapoints, and the *best fit* model parameters are defined as the parameters that, when plugged into the likelihood, maximize it. In other words, the likelihood is the product of all N of the individual datapoints' joint probability distributions. In practice, rather than choosing to *maximize* \mathcal{L} to determine the best fit, it is much more common to *minimize* $-2 \ln \mathcal{L}$ (or some proportion thereof) in order to determine the best fit model parameters, for reasons of computational flexibility. In the simplifying “traditional” case of no error bars in x and no slop whatsoever, $-2 \ln \mathcal{L}$ is equivalent to the χ^2 “goodness-of-fit” statistic, $\chi^2 = \sum_{n=1}^N [(y - y_c(x; \vartheta_m)) / \sigma_{y,n}]^2$. As such, in our general case, $-2 \ln \mathcal{L}$ is analogous to χ^2 , and has the form of

$$\begin{aligned} -2 \ln \mathcal{L} &= -2 \sum_{n=1}^N \ln p_n(\vartheta_m, \sigma_x, \sigma_y | x_n, y_n, \sigma_{x,n}, \sigma_{y,n}) \\ &= \sum_{n=1}^N \frac{[y_n - y_{t,n} - m_{t,n}(x_n - x_{t,n})]^2}{m_{t,n}^2 \Sigma_{x,n}^2 + \Sigma_{y,n}^2} - 2 \sum_{n=1}^N \ln \left(\frac{du_n}{dx} \frac{1}{\sqrt{m_{t,n}^2 \Sigma_{x,n}^2 + \Sigma_{y,n}^2}} \right) + C, \end{aligned} \quad (2.16)$$

following Equations (2.7) and (2.15), where C is a constant³.

Recall that the rotated coordinate system (u_n, v_n) in which the 1D model curve $\delta(v_n, v_{c,n}(u_n; \vartheta_m))$ is defined can be chosen at will, including the usage of different coordinates for different datapoints. As will be described in §6.1, different choices of these coordinates/of $\frac{du_n}{dx}$ will give different statistics, with noticeably different properties. The following section will show how a certain choice of these coordinates will lead to the TRK statistic.

2.2.3. The TRK Likelihood

As shown in Equation (2.16), The arbitrary choice of the rotated coordinates (u_n, v_n) and therefore the factor $\frac{du_n}{dx}$ will be what defines a given statistic. While various choices for $\frac{du_n}{dx}$ that lead to different statistics with different properties will be explored in §6.1, for now I will only examine the choice that leads to the TRK statistic, which is advantageous over other such statistics for reasons that will be addressed in §6.1.

³The explicit form of the constant C is given in Trotter (2011), which isn't necessary for the purposes of this paper, as constant offsets are arbitrary for the process of likelihood maximization.

2. The TRK Statistic

The TRK statistic is defined such that for some n^{th} datapoint, u_n is chosen to be perpendicular to the line segment connecting the centroid of the datapoint (x_n, y_n) with the tangent point $(x_{t,n}, y_{t,n})$ discussed in the previous section. This choice results in a likelihood of the form

$$\begin{aligned}\mathcal{L}^{\text{TRK}} &\propto \prod_{n=1}^N \sqrt{\frac{m_{t,n}^2 \Sigma_{x,n}^2 + \Sigma_{y,n}^2}{m_{t,n}^2 \Sigma_{x,n}^4 + \Sigma_{y,n}^4}} \exp \left\{ -\frac{1}{2} \frac{[y_n - y_{t,n} - m_{t,n}(x_n - x_{t,n})]^2}{m_{t,n}^2 \Sigma_{x,n}^2 + \Sigma_{y,n}^2} \right\} \\ -2 \ln \mathcal{L}^{\text{TRK}} &= \sum_{n=1}^N \frac{[y_n - y_{t,n} - m_{t,n}(x_n - x_{t,n})]^2}{m_{t,n}^2 \Sigma_{x,n}^2 + \Sigma_{y,n}^2} - \sum_{n=1}^N \ln \left(\frac{m_{t,n}^2 \Sigma_{x,n}^2 + \Sigma_{y,n}^2}{m_{t,n}^2 \Sigma_{x,n}^4 + \Sigma_{y,n}^4} \right) + C,\end{aligned}\tag{2.17}$$

which is the central equation of this work. One important property of the TRK statistic is that *it is essentially a one-dimensional χ^2 -like statistic that is measured in the direction of the tangent point*⁴. For a visualization of the geometry of the TRK statistic, see Figure 2.5. Hereafter, I will sometimes use the shorthand $\chi_{\text{TRK}}^2 \equiv -2 \ln \mathcal{L}^{\text{TRK}}$, especially in Chapter 4 where it is frequently used.

⁴The derivation of this is beyond the scope of this work, but is found in Trotter (2011).

2. The TRK Statistic

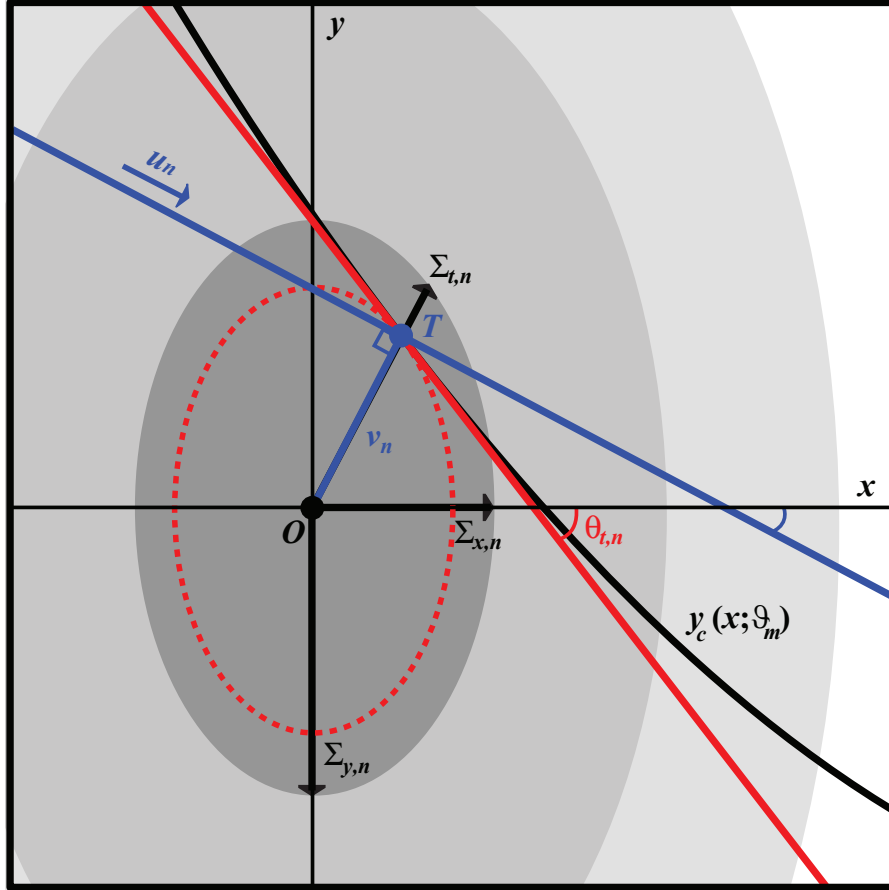


Figure 2.5.: Illustration of the geometry of the TRK statistic modified from Trotter (2011), given some datapoint and model. The datapoint is centered at (x_n, y_n) (point O), with convolved error ellipse described by the widths $(\Sigma_{x,n}, \Sigma_{y,n})$ (Equation (2.9)). The model curve $y_c(x; \vartheta_m)$ is tangent to the convolved error ellipse at tangent point $(x_{t,n}, y_{t,n})$ (point T), and the red line is the linear approximation of the model curve, with slope $m_{t,n} = \tan \theta_{t,n}$. The blue line indicates the rotated coordinate axis u_n for the TRK statistic, perpendicular to the v_n axis.

3. Properties of the TRK Statistic

3.0.1. Invertibility

A useful property for any two-dimensional statistic is that it is *invertible*, i.e. that if fitting a dataset's y vs x data gives some model curve $y_c(x)$, fitting x vs. y gives the inverse $x_c(y) = y_c(x)^{-1}$. In the Bayesian formalism, a statistic is invertible if running these two inverted fits yields the same likelihood function (Trotter (2011)). While mainly known to be used as a measure of the linear correlation of a dataset, one metric of invertibility is actually the ubiquitously-used Pearson Correlation Coefficient, R^2 , of Pearson (1896). To see this, consider some linear model with slope m_{yx} that was obtained by fitting to some y vs. x data. Similarly, fitting x vs. y for the same dataset gives some model line with slope m_{xy} . As shown in Trotter (2011), the correlation coefficient can then be found as $R^2 \equiv m_{xy}m_{yx}$; therefore, if the statistic used to fit is invertible, and therefore $m_{xy} = 1/m_{yx}$, we have that $R^2 = 1$. Proved by Trotter (2011), the TRK statistic is completely invertible. Therefore, by definition $R^2 = 1$ always for the TRK statistic, meaning that fitting results can always be trusted under inversion.

3.0.2. Scalability

Another important property of any statistic, although not immediately obvious, is its *scalability*. Here, a statistic is defined to be *scalable* if re-scaling the data along the x – or y – axis does not change the best fit arrived at from maximizing the likelihood. If a statistic is not scalable, then the best fit will *depend* on the choice of units of measurement, which can easily create unwanted behavior when fitting (see e.g. Trotter (2011)), given that there is usually never any *a priori* reason to choose some set of units over another.

3. Properties of the TRK Statistic

To examine the scalability of the TRK statistic, we will begin by noting that the statistic is invariant if both x - and y - axes are rescaled by the same factor, shown in Trotter (2011). As such, any rescaling that potentially affects the TRK statistic can always be defined as rescaling only the y -axis by some numerical factor s ¹. In order to see the effect of data rescaling within the TRK statistic, I multiply all y -axis dependent terms by some s within the TRK likelihood of Equation (2.17), which gives

$$\begin{aligned} \mathcal{L}_s^{\text{TRK}} &\propto \prod_{n=1}^N \sqrt{\frac{m_{t,n}^2 \Sigma_{x,n}^2 + \Sigma_{y,n}^2}{m_{t,n}^2 \Sigma_{x,n}^4 + s^2 \Sigma_{y,n}^4}} \exp \left\{ -\frac{1}{2} \frac{[y_n - y_{t,n} - m_{t,n}(x_n - x_{t,n})]^2}{m_{t,n}^2 \Sigma_{x,n}^2 + \Sigma_{y,n}^2} \right\} \\ &\propto \mathcal{L}^{\text{TRK}} \quad \text{for a fixed } s. \end{aligned} \quad (3.1)$$

As such, the standalone TRK statistic is *not* scalable, so different choices of scale will result in different best fit model parameters, including slop/extrinsic scatter (σ_x, σ_y) . Not only this, but it is impossible to determine anything about the relative fitness of best fits solely given numerical values of the likelihood function; what this means in practice is that the scaling factor s can not be fit to as a model parameter (Trotter (2011)). However, we will show in the following section that there *is* a way to quantitatively compare TRK fits done at different scales, so that this hurdle can be negated.

3.0.3. The TRK Correlation Coefficient

To begin, consider two TRK best fits gained from maximizing the likelihood (Equation (3.1)) at different scales (i.e. different values for s), given some model and dataset. Because the TRK statistic is completely invertible, the Pearson Correlation Coefficient R^2 is 1 for both fits. As such, in order to compare the two fits, a new correlation coefficient needs to be defined that can quantify the variance of the statistic's predictions between them. By convention, the new coefficient should follow similar properties to R^2 , insofar that it is restricted to the range of $[0, 1]$, and that it equals 1 if the two best fit lines being compared have the same slope (plotted in the same scale space).

I will begin with the case of linear fits, and continue on to generalize to arbitrary non-linear

¹Equivalently we can define $s \equiv s_y/s_x$, where s_y is the standalone rescaling of the y -axis, and s_x is the same for the x -axis.

3. Properties of the TRK Statistic

models. Trotter (2011) introduced a new correlation coefficient R_{TRK}^2 that is a function of the *difference* of the slopes of the models, rather than the *ratio*, as opposed to the Pearson R^2 (see §3.0.1). The *scale-dependent* TRK correlation coefficient is defined as

$$R_{\text{TRK}}^2(a, b) \equiv \tan^2 \left(\frac{\pi}{4} - \frac{|\theta_a - \theta_b|}{2} \right). \quad (3.2)$$

given a linear fit at $s = a$ with slope $m_a = \tan \theta_a$, and another at $s = b$ with slope $m_b = \tan \theta_b$ ². Note that these fits, although performed at different scales, have their angles compared within the original, $s = 1$ space. Clearly, if the two lines have the same slopes, $R_{\text{TRK}}^2 = 1$ as desired, and if the two lines differ in slope angle by 90° , i.e. they are orthogonal, $R_{\text{TRK}}^2 = 0$. Now that the difference between TRK fits at different scales can be compared, how do we determine the best scale at which to run a fit?

Consider how rescaling will affect the slop parameters σ_x and σ_y , i.e. how the total slop is distributed between these two parameters. Trotter (2011) showed that in the limit of slop-dominated data (i.e. arbitrarily small/zero error bars $\{\sigma_{x,n}, \sigma_{y,n}\}$ as compared to the extrinsic scatter/slop), $s \rightarrow 0$, $\sigma_x \rightarrow 0$; similarly, as $s \rightarrow \infty$, $\sigma_y \rightarrow 0$. This behavior occurs because as the scale s of the dataset is changed, the distribution of the total slop between σ_x and σ_y is correspondingly affected. This range of $s \in [0, \infty)$ is considered to be the *physically meaningful* range of fits. In the case of a dataset with non-zero error bars, this physically meaningful range becomes some subset interval $[a, b] \subset [0, \infty)$, where the number a is described as the *minimum scale*, while b is the *maximum scale*, as $\lim_{s \rightarrow a^+} \sigma_x = 0$ and $\lim_{s \rightarrow b^-} \sigma_y = 0$ (from Trotter, 2011)³. As any fits done outside of this interval are inherently unphysical, there must be some optimum $s_0 \in [a, b]$ that is the best scale at which to run a fit⁴.

In order to determine the optimum scale s_0 , the following iterative approach defined within Trotter (2011) is used. To begin, the first approximation of s_0 , $s_0^{(1)}$, is found to be the scale at which

$$R_{\text{TRK}}^2(a, s_0^{(1)}) = R_{\text{TRK}}^2(s_0^{(1)}, b) \equiv R_{\text{TRK}}^2. \quad (3.3)$$

² R_{TRK}^2 compares the *angles* (off of the x -axis) (θ_a, θ_b) of the lines rather than the *slopes* (m_a, m_b) for numerical efficacy, given that the former are restricted to the range of $(-\frac{\pi}{2}, \frac{\pi}{2})$, while the latter can be anywhere within $(-\infty, \infty)$.

³Here, I've taken the signs of the limits to indicate the direction of one-sided approach.

⁴By “unphysical”, I mean that such scales require imaginary best fit slops, i.e. $(\sigma_x^2, \sigma_y^2) < 0$ (Trotter, 2011).

3. Properties of the TRK Statistic

From here, we shift from the $s = 1$ space to this $s = s_0^{(1)}$ space, where the angles of the lines follow the transformation $\theta \rightarrow \arctan(s_0^{(1)} \tan \theta)$. The analysis of Equation (3.3) is then repeated in this new space to determine the next approximation for the optimum scale, $s_0^{(2)}$, i.e. finding the $s_0^{(2)}$ such that, e.g. in the case of a linear model,

$$\begin{aligned} R_{\text{TRK}}^2 &\equiv \tan^2 \left(\frac{\pi}{4} - \frac{\left| \arctan(s_0^{(1)} \tan \theta_a) - \arctan(s_0^{(1)} \tan \theta_{s_0^{(2)}}) \right|}{2} \right) \\ &= \tan^2 \left(\frac{\pi}{4} - \frac{\left| \arctan(s_0^{(1)} \tan \theta_{s_0^{(2)}}) - \arctan(s_0^{(1)} \tan \theta_b) \right|}{2} \right), \end{aligned} \quad (3.4)$$

where $\theta_{s_0^{(2)}}$ is the position angle of the best-fit line at scale $s_0^{(2)}$, as measured in $s = 1$ space. From here, we set $s_0^{(2)} \rightarrow s_0^{(1)}$, and repeat until convergence to the final value of s_0 . It is at this optimum scale that we actually run fits, compute model parameter uncertainties (see §4.4.1), etc. The details of how Equation (3.3) is solved in practice to determine s_0 are given in §4.3.

The TRK correlation coefficient as given in Equation (3.2) can only be used for linear models. As such, Trotter (2011) presented a logical generalization to nonlinear models, as the average of the differences of the slope angles at all N tangent points at two scales a and b :

$$R_{\text{TRK}}^2(a, b) \equiv \frac{1}{N} \sum_{n=1}^N \tan^2 \left(\frac{\pi}{4} - \frac{|\theta_{t,n;a} - \theta_{t,n;b}|}{2} \right). \quad (3.5)$$

Here, $\theta_{t,n;a} = \arctan m_{t,n;a}$ and $\theta_{t,n;b} = \arctan m_{t,n;b}$ are the position angles of the best-fit curves at the tangent point to the n^{th} datapoint at scales a and b , respectively. This expression for R_{TRK}^2 can then be used to determine s_0 using the same method described by the previous section and Equation (3.3); in this case then, Equation (3.4) becomes

$$\begin{aligned} R_{\text{TRK}}^2 &\equiv \frac{1}{N} \sum_{n=1}^N \tan^2 \left(\frac{\pi}{4} - \frac{\left| \arctan(s_0^{(1)} \tan \theta_{t,n;a}) - \arctan(s_0^{(1)} \tan \theta_{s_0^{(2)}}) \right|}{2} \right) \\ &= \frac{1}{N} \sum_{n=1}^N \tan^2 \left(\frac{\pi}{4} - \frac{\left| \arctan(s_0^{(1)} \tan \theta_{s_0^{(2)}}) - \arctan(s_0^{(1)} \tan \theta_{t,n;b}) \right|}{2} \right). \end{aligned} \quad (3.6)$$

With this, we have covered all of the foundations and properties of the TRK statistic that

3. Properties of the TRK Statistic

are needed to describe how TRK fits are completed in practice. In the next chapter, I will delve into the suite of algorithms that I created to perform fitting, scale optimization, model parameter distribution generation, and other core fitting algorithms.

4. The TRK Codebase: Core Algorithms

The previous chapter introduced all of the formalism needed to use the TRK statistic in principle, but how can TRK fits be done in practice? What are the implementational details? What options and configurations are available when using the TRK statistic? While the statistic itself was created by Trotter (2011), it was only implemented in the form of a genetic algorithm-based scientific codebase for testing. In order to have a production-quality codebase for the usage of TRK, I created a new suite of algorithms from scratch in C++, that completely overhauls how TRK fits are computed, including many additions and optimizations as compared to the previous codebase. With this new codebase, TRK can be used in a fully customizable, yet easy-to-use manner, with a host of options. The code and full documentation can be downloaded at <https://github.com/nickk124/TRK>.

The first part of this chapter will discuss the implementation of the central part of the TRK statistic, the TRK likelihood function \mathcal{L}^{TRK} of Equation (2.17), that is maximized to obtain best fits. Here, I will explore my the algorithms that I use to determine the tangent points described by Equation (2.13), and maximize \mathcal{L}^{TRK} with respect to all model parameters—including slop—to obtain best fits. Following this, I will delve into the routine that is used to determine the optimum fitting scale s_0 , using the formalism described in §3.0.3. From there, I will explore the Monte Carlo methods used to generate the full probability distributions and uncertainties of model parameters.

4.1. Tangent Point Finding

As shown in §2.2.2 and §2.2.3, for a given model curve $y_c(x; \vartheta_m)$ and datapoint (x_n, y_n) , the point where the convolved error ellipse of the datapoint described by the convolved error parameters $(\Sigma_{x,n}, \Sigma_{y,n})$ of Equation (2.9) is tangent to the model curve, $(x_{t,n}, y_{t,n})$, is a central part of the TRK likelihood \mathcal{L}^{TRK} (Equation (2.17)). Any time that the likelihood must be computed for a new set of model and slop parameters, all N of these tangent points must be re-computed, and efficiently.

Recall that in order to determine such a tangent point $x_{t,n}$ (and therefore $y_{t,n} = y_c(x_{t,n}; \vartheta_m)$), Equation (2.13) must be solved implicitly for $x_{t,n}$ in the general case of some nonlinear model. In more practical terms, this means that the root(s) along the x -axis of the left hand side of Equation (2.13) are such tangent point(s). To determine these tangent points, I use (a slightly modified version of) the Two-Point Newton-Raphson algorithm created by Tiruneh, Ndlela, and Nkambule (2013) to find the roots of Equation (2.13). This algorithm has a number of benefits over other root-finding methods such as bisection and the traditional Newton-Raphson method, especially when dealing with complicated nonlinear functions that otherwise give convergence and speed issues. My pseudocode implementation of it is shown in Algorithm 1.

It is essential to note that for certain non-monotonic, nonlinear models, there can easily be multiple tangent points for a given datapoint; see Figure 4.1 as an example. In this case, I take the tangent point that maximizes the joint posterior probability¹ as the tangent point to be used when evaluating the likelihood. However, only using the rootfinder of Algorithm 1 once per datapoint is insufficient, as some initial guess for the algorithm will always only return the same, single tangent point. In order to reliably determine *all* possible tangent points for a given datapoint, to properly maximize the likelihood, I use a logical routine described in Algorithm 2. The goal of this routine is to determine various initial guesses to supply to the root-finder until all possible tangent point-finding options have been exhausted. To begin this algorithm, the root finder is used with the default initial guess to determine the first tangent point. From here, I use a quadratic approximation (of Equation (2.13)) through this first tangent point to approximate up to two more tangent

¹I.e. for some n^{th} datapoint, the joint posterior probability is the n^{th} term in the product of Equation (2.17).

Algorithm 1: Modified Two-Point Newton-Raphson algorithm for finding a single tangent point.

```

1 Function TwoPointNR
   Input : Model parameters  $\vartheta_m$ , datapoint  $(x_n, y_n)$ , its convolved errors
             $(\Sigma_{x,n}, \Sigma_{y,n})$ , and initial tangent point guess of  $x_{\text{guess}}$ .
   Output:  $x_{t,n}$ 
2 begin
3   Initialize guess of  $x_{k-1} = x_{\text{guess}}$  and  $x_k = x_{\text{guess}} + \Sigma_{x,n}/\sqrt{10}$ 
4   while not converged do
5     Main Two-Pt NR loop:
6      $y_{k-1} = [y_c(x_{k-1}; \vartheta_m) - y_n] \frac{dy_c(x_{k-1}; \vartheta_m)}{dx} \Sigma_{x,n}^2 + (x_{k-1} - x_n) \Sigma_{y,n}^2$ 
7      $y_k = [y_c(x_k; \vartheta_m) - y_n] \frac{dy_c(x_k; \vartheta_m)}{dx} \Sigma_{x,n}^2 + (x_k - x_n) \Sigma_{y,n}^2$ 
8      $\frac{dy_k}{dx} = \left( \frac{dy_c(x_k; \vartheta_m)}{dx} \right)^2 + [y_c(x_k; \vartheta_m) - y_n] \frac{d^2 y_c(x_k; \vartheta_m)}{dx^2} \Sigma_{x,n}^2 + \Sigma_{y,n}^2$ 
9      $r = 1 - \frac{y_k}{y_{k-1}} \left( \frac{y_k - y_{k-1}}{x_k - x_{k-1}} \middle/ \frac{dy_k}{dx} \right)$ 
10     $x_{k+1} = \left( 1 - \frac{1}{r} \right) x_{k-1} + \frac{x_k}{r}$ 
11     $x_{k-1} = x_k, \quad x_k = x_{k+1}$ 
12  end
13  return  $x_{t,n} = x_k$ 
14 end

```

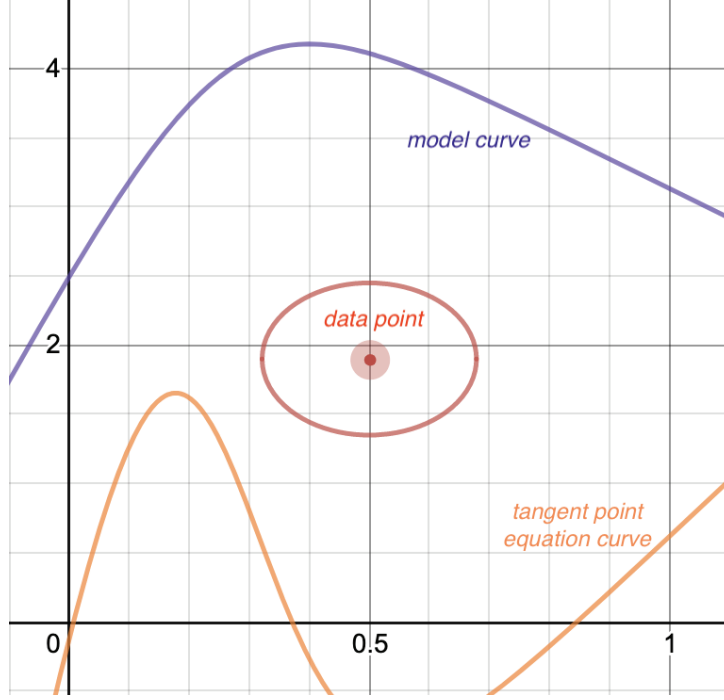


Figure 4.1.: Example of a model curve $y_c(x; \vartheta_m)$ (purple, top) and datapoint (x_n, y_n) with convolved error ellipse (red, middle) described by $(\Sigma_{x,n}, \Sigma_{y,n})$ where there are multiple points where y_c is tangent to the ellipse. This translates to there being multiple solutions/roots to/of Equation (2.13), which is plotted in orange at the bottom.

points, and use these two approximate points as initial guesses for the root finder, which will determine the remaining roots². In general, I have found that non-periodic functions generally have no more than three tangent points for a given datapoint. Finally, I note that in the current C++ implementation of the TRK suite, the option to parallelize the determination of all N tangent points for a given evaluation of \mathcal{L}^{TRK} is provided³.

4.2. Likelihood Maximization

The best fit at some scale s is defined to be the model parameters ϑ_m and slop parameters (σ_x, σ_y) that maximize the TRK likelihood \mathcal{L}^{TRK} of Equation (2.17)⁴; however, as described in §2.2.2, in practice the TRK suite *minimizes* $\chi_{\text{TRK}}^2 \equiv -2 \ln \mathcal{L}^{\text{TRK}}$ to determine best fits. For less general statistics where slop is not included in the likelihood, $-2 \ln \mathcal{L}$ is often

²I also implemented a few modifications to this to account for certain edge cases (see Algorithm 2).

³Parallelizing the tangent-point finder isn't always advisable for simple, linear models, as the two-point Newton-Raphson routine runs so fast with these models that the computational overhead for starting and stopping each tangent point's computational thread is greater than the power needed to actually find the tangent point itself. As such, this feature is most useful for nonlinear models.

⁴The method used to determine the optimum fitting scale s_0 will be covered in §4.3.

Algorithm 2: Find all possible tangent points for a given model and datapoint.

```

1  Function FindAllTangentPoints
   Input : Model parameters  $\vartheta_m$ , datapoint  $(x_n, y_n)$ , and its convolved errors
            $(\Sigma_{x,n}, \Sigma_{y,n})$ .
   Output: Array of all tangent points  $\{x_{t,n}^i\}$ .

2  begin
3      Initialize empty  $\{x_{t,n}^i\}$ , and initialize  $x_{\text{guess},0} = x_n$ .
4      while not all tangent points  $x_{t,n}^i$  found do
5          First tangent point is the one that Two-Point Newton Raphson finds:
6          Append  $x_{t,n}^{\text{new}} = \text{TwoPointNR}(x_{\text{guess}} = x_{\text{guess},0})$  to  $\{x_{t,n}^i\}$ 
7          if New root found is same as one from previous iteration then
8              break
9          end
10         if TwoPointNR oscillating between two tangent points  $x_1, x_2 \in \{x_{t,n}^i\}$ 
            then
11              $x_{\text{mid}} = \text{TwoPointNR}[x_{\text{guess}} = \text{Midpoint}(x_1, x_2)]$  to  $\{x_{t,n}^i\}$ 
12              $\{x_{t,n}^i\} = \{x_1, x_2, x_{\text{mid}}\}$ 
13             break
14         end
15         Use quadratic approximation through  $x_{t,n}^{\text{new}}$  to approximate two more
            tangent points if possible (see Algorithm 3):
16          $\{x_{t,n}^{\text{approx}}\} = \text{QuadraticApprox}(x_{t,n}^{\text{new}})$ 
17         if  $\{x_{t,n}^{\text{approx}}\}$  has 2 additional, approximate tangent points  $x_{\text{guess},1}, x_{\text{guess},2}$ 
            then
18             Let  $(x_{\text{guess},1}, x_{\text{guess},2})$  = these two new approximate tangent points
19             if  $x_{t,n}^{\text{new}} \in \{x_{\text{guess},1}, x_{\text{guess},2}\}$  then
20                  $x_1 = \text{TwoPointNR}(x_{\text{guess}} = x_{\text{guess},1})$ 
21                  $x_2 = \text{TwoPointNR}(x_{\text{guess}} = x_{\text{guess},2})$ 
22                  $\{x_{t,n}^i\} = \{x_1, x_2, x_{t,n}^{\text{new}}\}$ 
23                 break
24             end
25             else
26                  $x_{\text{guess},0} = \text{Median}(x_{t,n}^{\text{new}}, x_{\text{guess},1}, x_{\text{guess},2})$ 
27             end
28         end
29         if  $\{x_{t,n}^{\text{approx}}\}$  contains no new tangent points then
30              $x_{\text{min}} = \min\{x_n\}$ ,  $x_{\text{max}} = \max\{x_n\}$ 
31              $x_{\text{left}} = \text{TwoPointNR}(x_{\text{guess}} = x_{\text{min}})$ 
32              $x_{\text{right}} = \text{TwoPointNR}(x_{\text{guess}} = x_{\text{max}})$ 
33             Append  $x_{\text{left}}, x_{\text{right}}$  to  $\{x_{t,n}^i\}$ 
34             break
35         end
36         else if Two tangent points  $x_1, x_2$  found in total, but  $\{x_{t,n}^{\text{approx}}\}$  contains
            no new points then
37              $\{x_{t,n}^i\} = \{x_1, x_2\}$ 
38             break
39         end
40     return  $\{x_{t,n}^i\}$ 
41 end
    
```

Algorithm 3: Use a quadratic approximation about a found tangent point to determine guesses for any additional unknown tangent points.

```

1 Function QuadraticApprox
   Input : Model parameters  $\vartheta_m$ , datapoint  $(x_n, y_n)$ , its convolved errors
             $(\Sigma_{x,n}, \Sigma_{y,n})$ , and tangent point  $x_{t,n}$  determined by Two-Point
            Newton Raphson root-finder.
   Output: Approximations of additional unknown tangent points  $\{x_{t,n}^{\text{approx}}\}$ , if
            any remain.
2 begin
3   Initialize empty array of possible approximate tangent points  $\{x_{t,n}^{\text{approx}}\}$ 
4   Approximate
      
$$y_c(x) \approx y_c(x_{t,n}) + \frac{dy_c(x_{t,n})}{dx} (x - x_{t,n}) + \frac{1}{2} \frac{d^2 y_c(x_{t,n})}{dx^2} (x - x_{t,n})^2$$

5   Use this approximation for  $y_c$  with Equation (2.13) to get a cubic equation
      for  $\{x_{t,n}^{\text{approx}}\}$ .
6   if The discriminant of this cubic equation  $> 0$  then
7     There exists two real and distinct additional approximate tangent
      points:
8     Use a cubic equation solver to analytically determine the two
      additional approximate tangent points, and append them to  $\{x_{t,n}^{\text{approx}}\}$ .
9   end
10  else
11    There exists no additional real approximate tangent points; the initial
    point  $x_{t,n}$  found is the only one.
12  end
13  return  $\{x_{t,n}^{\text{approx}}\}$ 
14 end

```

4. The TRK Codebase: Core Algorithms

minimized using methods such as the Gauss-Newton algorithm, e.g. Maples et al. (2018). The Gauss-Newton algorithm is usually quick to converge because it uses the partial derivatives of the model function with respect to the model parameters. However, because an explicit expression is required for each derivative, this method does *not* translate over to the TRK statistic, as the dependence of the model curve y_c on the slop parameters (σ_x, σ_y) cannot be explicitly written down in the general case. Because χ_{TRK}^2 must be minimized not only with respect to the model parameters, but also the slop parameters, Gauss-Newton, or any other method that requires explicit parameter derivatives, is not an option.

Instead, the TRK suite uses a modified version of the Downhill Simplex algorithm of Nelder and Mead (1965) to minimize χ_{TRK}^2 , with explicit implementation given in Algorithm 4. The basic mechanism of this algorithm is that given \mathcal{M} model and slop parameters, a *simplex*⁵ with $\mathcal{M} + 1$ vertices is initialized and evolves within parameter space along the surface of χ_{TRK}^2 , with its evolution depending on the values of χ_{TRK}^2 at its various vertices, in order to find the minimum of χ_{TRK}^2 . This method is advantageous due to its general reliability, and the fact that no matter the number of parameters in the model, the only function that the user need supply to the algorithm is the model curve⁶, as opposed to an \mathcal{M} -parameter model requiring the user to supply \mathcal{M} partial derivatives of the model, as in the Gauss-Newton algorithm⁷. For my implementation of the Nelder-Mead algorithm, I also made the following modifications:

1. Any user-supplied prior probability distributions that place hard upper and/or lower bounds on model parameters will act as “walls” for the evolution of the simplex, i.e. if the simplex enters a region of a forbidden value of a model parameter, χ_{TRK}^2 will evaluate as numerical infinity.
2. Once the best fit model and slop parameters have been determined by the minimization of χ_{TRK}^2 , small values of slop are “pegged” to zero if they are within a given tolerance, for the usage of the scale optimization algorithm described in §4.3.

⁵A simplex is essentially a higher-dimensional generalization of a triangle.

⁶However, the user still needs to supply the first two x -derivatives of y_c , for the tangent point-finding routine of §4.1.

⁷We note that for complicated nonlinear and/or high dimensional models, the simplex method can be fairly dependent on the user-supplied initial guess for the model and slop parameters, given that such models are often fraught with local minima.

Algorithm 4: Modified Nelder-Mead Downhill Simplex algorithm for determining best fits according to $\chi_{\text{TRK}}^2 \equiv \mathcal{L}^{\text{TRK}}$. Note that if a simplex vertex v enters a region of parameter space forbidden by any provided bounded parameter priors, $\chi_{\text{TRK}}^2(v) \sim +\infty$.

```

1 Function DownhillSimplex
    Input : Model  $y_c$ , initial guess for model and slop parameters
              $\{\vartheta_m, (\sigma_x, \sigma_y)\}^{\text{guess}}$  of dimension  $\mathcal{M}$ , dataset  $\{x_n, y_n\}$  with error bars
              $\{\sigma_{x,n}, \sigma_{y,n}\}$ , given some fit scale  $s$ 
    Output: Best fit parameters  $\{\vartheta_m, (\sigma_x, \sigma_y)\}$ 
2 begin
3     Initialize simplex  $\Delta$  with  $\mathcal{M} + 1$  vertices  $v_i, i \in \{0, \dots, \mathcal{M}\}$  about
        $v_0 = \{\vartheta_m, (\sigma_x, \sigma_y)\}^{\text{guess}}$ , with values of  $\chi_{\text{TRK}}^2(v_i) \equiv \chi_i^2$  at each  $v_i$ 
4     while  $\text{StdDev}\{\chi_i^2\} > \text{tolerance}$  do
5         Each iteration of this is one simplex step.
6         Reorder the the vertices  $v_i$  of  $\Delta$  with respect to ascending ordering of
            $\chi_{\text{TRK}}^2(v_i) \equiv \chi_i^2$ .
7         Compute centroid excluding “worst” vertex  $v_{\mathcal{M}}$ ,  $c \equiv \frac{1}{\mathcal{M}} \sum_{i=0}^{\mathcal{M}-1} v_i$ .
8         Reflect: Compute reflection point of  $v_{\mathcal{M}}$ ,  $v_r \equiv c + \alpha(c - v_{\mathcal{M}})$ 
9         if  $\chi_0^2 \leq \chi_r^2 < \chi_{\mathcal{M}-1}^2$  then
10              $v_{\mathcal{M}} \rightarrow v_r$ 
11             continue
12         end
13         if  $\chi_r^2 < \chi_0^2$  then
14             Expand( $\Delta$ ) (see Algorithm 10 in §A)
15             continue
16         end
17         if  $\chi_r^2 \geq \chi_{\mathcal{M}-1}^2$  then
18             Contract( $\Delta$ ) (see Algorithm 10 in §A)
19             continue
20         end
21         Shrink( $\Delta$ ) (see Algorithm 10 in §A)
22     end
23     pegSlopToZero( $\Delta$ )
24     makeSlopPositive( $\Delta$ ) (see bullet point 2 on pg. 27)
25     return Best fit  $\{\vartheta_m, (\sigma_x, \sigma_y)\} \equiv v_{\mathcal{M}}$ 
26 end
    
```

4. The TRK Codebase: Core Algorithms

Additionally, any negative best fit slop values (σ_x, σ_y) are made positive for the final best fit, because although the simplex explores parameter space with respect to (σ_x, σ_y) , these terms are always squared when they appear in the likelihood (Equation (2.17)), so a fitted negative value is equivalent to a positive.

4.3. Scale Optimization

As described in §3.0.2, the TRK statistic/likelihood is *not* scalable by default, i.e. multiplying a dataset along the y -axis by some numerical factor s will result in different best fits for various values of s . As such, we are presented with the problem of determining the *global* optimum scale $s = s_0$ to perform fits for a given dataset and model. Such a method was explored schematically in §3.0.3 on pg. 18 as presented by Trotter (2011), and in the following section I will present the explicit implementational details of the *scale optimization algorithm*.

Given some dataset and model, from Equation (3.3), we must begin by determining the *minimum* and *maximum* fitting scales a and b as described in §3.0.3. Recall from this section that a is the scale where as $s \rightarrow a^+$, the slop in the x -direction, $\sigma_x \rightarrow 0$. Similarly, b is the scale where as $s \rightarrow b^-$ the y -slop $\sigma_y \rightarrow 0$. An example of this behavior is presented in Figure 4.2, where I plot best fit slop values σ_x, σ_y for various values of s , given a linear model⁸.

In order to determine the scale extrema a and b for a given model and dataset, I implemented a type of bracketing/bisection method that, by running fits at various scales (using the Downhill Simplex routine of Algorithm 4) and observing best fit slop values, finds the exact scales where the two slops go to zero. This method determines a first, and then b ; I present the method for determining a as Algorithm 5, and the similar method for determining b is given as Algorithm 11 in §A. Note that I also make several modifications for improved efficiency, such as using best fit values for σ_y obtained from determining a to start the routine for determining b at a better initial guess. I also provide the option

⁸Specifically, for the c_1 vs. c_2 spectral dust extinction model and dataset described in §6.2.

4. The TRK Codebase: Core Algorithms

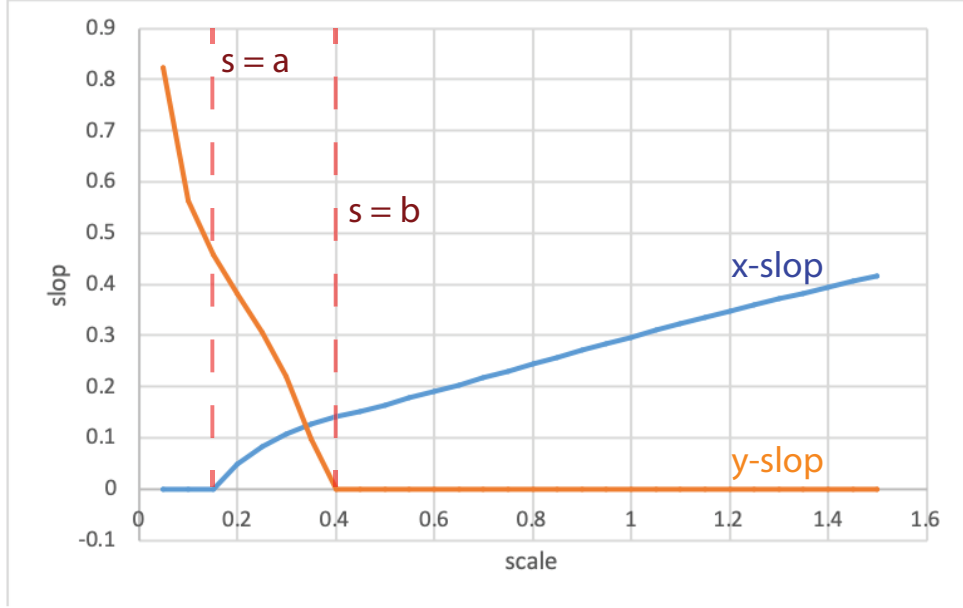


Figure 4.2.: Example of slop/extrinsic scatter (σ_x, σ_y) dependence on fitting scale s (for some linear model). Note that as is predicted in §3.0.3, there exist minimum and maximum scales $s = a$ and $s = b$ (respectively) such that $\lim_{s \rightarrow a^+} \sigma_x = 0$ and $\lim_{s \rightarrow b^-} \sigma_y = 0$.

to find the two scale extrema at the same time using parallel computing (in the current, C++ implementation of the TRK suite).

Now that the method for determining the scale extrema a and b for a certain model and dataset has been presented, the optimum scale s_0 needs to be determined. From §3.0.3, in order to determine successive approximations for s_0 until convergence, Equation (3.6) must be repeatedly solved for $s_0^{(2)}$ given the previous iteration's solution $s_0^{(1)}$. I begin this process with $s_0^{(1)} = (a + b)/2$, and then successively solve Equation (3.6) numerically while setting $s_0^{(1)} = s_0^{(2)}$ after each iteration.

In practice, Equation (3.6) is solved by rewriting it as

$$\begin{aligned} \tilde{R}_{\text{TRK}}^2(s_0^{(2)}; s_0^{(1)}, a, b) &\equiv \frac{1}{N} \sum_{n=1}^N \tan^2 \left(\frac{\pi}{4} - \frac{\left| \arctan(s_0^{(1)} \tan \theta_{t,n;a}) - \arctan(s_0^{(1)} \tan \theta_{s_0^{(2)}}) \right|}{2} \right) \\ &\quad - \frac{1}{N} \sum_{n=1}^N \tan^2 \left(\frac{\pi}{4} - \frac{\left| \arctan(s_0^{(1)} \tan \theta_{s_0^{(2)}}) - \arctan(s_0^{(1)} \tan \theta_{t,n;b}) \right|}{2} \right) \\ &= 0, \end{aligned} \tag{4.1}$$

and numerically solving the equation for $s_0^{(2)}$ given the previous $s_0^{(1)} = s_0^{(2)}$ using a bisection

Algorithm 5: Bracketing/Bisection-type method for determining minimum fitting scale a for some model and dataset.

```

1  Function FindMinimumScale
   Input : Model  $y_c$  and dataset  $\{x_n, y_n\}$  with error bars  $\{\sigma_{x,n}, \sigma_{y,n}\}$ .
   Output: Minimum fitting scale  $a$ .
2  begin
3      Determine brackets  $(l, r)$  for min scale  $a$ :
4      Initialize bisection brackets  $l = s = 0, r = s = 1$  and  $s_{\text{trial}} = s = 1$ .
5       $\sigma_x(s_{\text{trial}}) \leftarrow \text{DownhillSimplex}(s = s_{\text{trial}})$ 
6      Initialize step modifier  $\alpha = 0.5 \times s_{\text{trial}}$ .
7      if  $\sigma_x(s_{\text{trial}}) > 0$  then
8           $r = s_{\text{trial}}$ 
9           $l_{\text{trial}} = s_{\text{trial}}$ 
10          $\sigma_x(l_{\text{trial}}) = \text{DownhillSimplex}(s = l_{\text{trial}})$ 
11         while  $\sigma_x(l_{\text{trial}}) > 0$  do
12              $l_{\text{trial}} = l_{\text{trial}} - \alpha$ 
13              $\sigma_x(l_{\text{trial}}) = \text{DownhillSimplex}(s = l_{\text{trial}})$ 
14              $\alpha = 0.5 \times \alpha$ 
15              $r = l_{\text{trial}}$ 
16         end
17          $l = l_{\text{trial}}$ 
18     end
19     else if  $\sigma_x(s_{\text{trial}}) = 0$  then
20          $l = s_{\text{trial}}$ 
21          $r_{\text{trial}} = s_{\text{trial}}$ 
22          $\sigma_x(l_{\text{trial}}) = \text{DownhillSimplex}(s = l_{\text{trial}})$ 
23         while  $\sigma_x(r_{\text{trial}}) = 0$  do
24              $r_{\text{trial}} = r_{\text{trial}} + \alpha$ 
25              $\sigma_x(r_{\text{trial}}) = \text{DownhillSimplex}(s = r_{\text{trial}})$ 
26              $l = r_{\text{trial}}$ 
27         end
28          $r = r_{\text{trial}}$ 
29     end
30     Use bisection to determine  $a$  now that we have brackets  $(l, r)$ :
31      $a_{\text{trial}} = (l + r)/2$ 
32      $\sigma_x(a_{\text{trial}}) = \text{DownhillSimplex}(s = a_{\text{trial}})$ 
33     while  $|l - r| \geq \text{tolerance1}$  AND  $\sigma_x(a_{\text{trial}}) \geq \text{tolerance2}$  do
34          $a_{\text{trial}} = (l + r)/2$ 
35          $\sigma_x(a_{\text{trial}}) = \text{DownhillSimplex}(s = a_{\text{trial}})$ 
36         if  $\sigma_x(a_{\text{trial}}) > 0$  then
37              $r = a_{\text{trial}}$ 
38         end
39         else if  $\sigma_x(a_{\text{trial}}) = 0$  then
40              $l = a_{\text{trial}}$ 
41         end
42     end
43     return  $a = a_{\text{trial}}$ 
44 end
    
```

root-finding routine, repeating as necessary until convergence⁹. I then set the final optimum scale s_0 to be the last iteration of $s_0^{(2)}$. This algorithm is presented in its entirety as Algorithm 12 in §A, on pg. 79.

4.4. Model Parameter Distribution and Uncertainty Computation

4.4.1. Using Adaptive MCMC to Sample Parameter Distributions

As a review, so far I have covered everything that is needed to determine best fit model and slop parameters for any given nonlinear model and dataset with intrinsic and extrinsic two-dimensional uncertainties. In practice, I run the scale optimization routine (Algorithms 5, 11, and 12 sequentially) to determine the optimum fitting scale s_0 , and then find the best fit model and slop parameters $\{\vartheta_m, (\sigma_x, \sigma_y)\}$ using the Nelder-Mead downhill simplex to maximize the TRK likelihood function \mathcal{L}^{TRK} (Algorithm 4). In reality, the *uncertainties* of the best fit model parameters (and possibly the slop parameters) are also often desired as a result of a fit, if not their complete *posterior* probability distributions¹⁰. Furthermore, if any prior information is known about the model parameters in the form of *prior* probability distribution(s), it can be essential to introduce them to the computation of the posteriors, following Bayes' Theorem (Equation (2.2)).

There are a number of methods that can be used to sample a parameter's (posterior probability) distribution, but the types of methods that offer some of the most flexibility, speed, and support for priors are Markov Chain Monte Carlo, or *MCMC* methods. MCMC methods are a class of algorithms used to sample probability distributions, and they fall under the broad umbrella of *Monte Carlo Methods* which are, generally speaking, the usage of the ability of computers to rapidly generate random numbers to simulate

⁹Here I use bisection instead of other root-finding algorithms (e.g. Newton-Raphson) because most of these other algorithms require derivatives of the function with respect to the independent variable. As the derivative of Equation (4.1) with respect to $s_0^{(2)}$, for example, is ill-defined, the usage of bisection is a necessity. Furthermore, because this dependence of slop on scale has shown to be monotonic in all models that I have explored, bisection is perfectly suited for the task.

¹⁰The latter often when the distribution(s) of the parameter are not simply Gaussian.

4. The TRK Codebase: Core Algorithms

useful numerical results, that are often otherwise computationally intractable with more straight-forward methods. A simple example of a Monte Carlo method is the continuous sampling of a *proposal* Gaussian distribution; as the number of samples becomes larger, the *generated distribution* will converge to the proposal distribution. A *Markov Chain* can generally be described as a process that continually changes from state to state following certain transition probability rules that are dependent on the previous state. Together, the moniker *Markov Chain Monte Carlo* describes how a Markovian random walk is taken through parameter space to sample and generate the distribution of the parameters over time, according to chosen rules for the evolution and sampling. In turn, this sampling of the distribution can be used to estimate uncertainty, central tendency (e.g. mean, mode etc.), and other useful quantities. For the TRK algorithm suite, the specific MCMC method that we use is the adaptive version of the classic Metropolis-Hastings sampling algorithm of Hastings (1970), which will be described as follows¹¹.

To begin, recall from §2.1 that the *posterior probability distribution* of a model is the probability distribution of obtaining some values $\Theta \equiv \{\vartheta_m, (\sigma_x, \sigma_y)\}$ for the model (and slop) parameters given a dataset $D \equiv \{x_n, y_n, \sigma_{x,n}, \sigma_{y,n}\}$. As such, in order to quantify any uncertainty of the predictions laid out by the best fit model parameters, the posteriors need to be examined. From Bayes' Theorem (Equation (2.2)), then, sampling the posterior is *equivalent* to (besides a factor of proportionality¹²) sampling the likelihood function multiplied by any priors¹³, i.e.

$$P(\Theta|D) \propto \mathcal{L}^{\text{TRK}}(D|\Theta)p(\Theta). \quad (4.2)$$

Generally, the Metropolis-Hastings method works by iteratively generating a sequence of samples of parameters, that as the number of samples $R \rightarrow +\infty$, converges to the true probability distribution of the parameters. Given some sample Θ_i , the possible distribution

¹¹I chose Metropolis-Hastings sampling over other methods, e.g. Hamiltonian Monte Carlo, due to it's speed and because I have seen most model parameter distributions to be well behaved enough for Metropolis-Hastings to be sufficient for the purposes of this algorithm.

¹²Again, this constant factor is of no consequence because in most cases, we often don't care about the absolute probability of a some range of values for parameters, but rather the relative probability as compared to another range of possible values. Even if we did, we could integrate over the posterior to compute the constant of proportionality, as it is just a normalization constant found by the condition that $\int p(\Theta|D) d\Theta = 1$.

¹³Note that if no priors are given for some or all of the parameters, the prior distribution(s) are *uninformative*, or flat, i.e. $p(\Theta) = 1$, such that the posterior is then directly proportional to the likelihood.

4. The TRK Codebase: Core Algorithms

of the next potential sample Θ_t , or the *proposal distribution* $Q(\Theta_t|\Theta_i)$, is dependent on the value of Θ_i (which is why the sequence of samples is a Markov Chain). I note that by default, Q is implemented as a Gaussian distribution, which could easily be changed if needed. The acceptance of this next potential sample is dependent on the proposal distribution: if it is accepted, it becomes the next step in the Markov Chain; if not, the sample is discarded and another potential sample is generated. The probability of acceptance for a potential parameter sample Θ_t is based off of the ratio between the posterior evaluated at the potential sample, and that at the previous sample Θ_i ; specifically, Θ_t is accepted if $\alpha \geq u$, where $\alpha \equiv \frac{\mathcal{L}^{\text{TRK}}(\Theta_t)p(\Theta_t)}{\mathcal{L}^{\text{TRK}}(\Theta_i)p(\Theta_i)}$ and $u \sim \mathcal{U}(0, 1)$ (\mathcal{U} indicates the uniform distribution)¹⁴.

The proposal distribution $Q(\Theta_t|\Theta_i)$ is proportional to the target *true* posterior distribution $P(\Theta)$ (in our case we assume a Gaussian distribution for this, which can easily be changed), while the *location* (i.e. mean) of $Q(\Theta_t|\Theta_i)$ is what changes from sample to sample (specifically, $Q(\Theta_t|\Theta_i)$ is centered on the most recent sample on the chain Θ_i). The *size*, or in other words the *covariance matrix* Σ and therefore standard deviations of $Q(\Theta_t|\Theta_i)$ —the latter of which can be considered to be analogous to the step size(s) of the Markov Chain for each parameter, as will be shown shortly—on the other hand must be chosen prior to sampling, and will very much affect the quality and efficiency of the overall sampling.

To see this behavior, consider some parameter m that is Gaussian-distributed, that we wish to sample with the Metropolis-Hasting method. We need to choose the standard deviation/width σ_m of the MCMC proposal distribution $Q(m)$ for the sampler, which we will refer to as the “step size” of the Markov Chain. We will use a fairly large sample count of $R = 100,000$ ¹⁵, and run the sampler given three noticeably different values of σ_m . The results of these three separate samplings are shown in Figure 4.3; clearly, choosing

¹⁴I note that in regions of parameter space that evaluate to extremely high likelihood, this ratio of posteriors $\frac{P(\Theta_t)}{P(\Theta_i)} = \frac{\mathcal{L}^{\text{TRK}}(\Theta_t)p(\Theta_t)}{\mathcal{L}^{\text{TRK}}(\Theta_i)p(\Theta_i)}$ can evaluate with computational underflow or overflow errors. As such, in practice the TRK suite samples in log space, i.e. some trial Θ_t is accepted given a previous Θ_i if $\ln \alpha = \ln \mathcal{L}^{\text{TRK}}(\Theta_t) - \ln \mathcal{L}^{\text{TRK}}(\Theta_i) + \ln p(\Theta_t) - \ln p(\Theta_i) \geq \ln u$, where again $u \sim \mathcal{U}(0, 1)$; this helps combat such numerical errors (<https://stats.stackexchange.com/users/63677/forgottenscience>) (2020). (Observing Equation (2.17), these errors can occur due to having a large dataset, small error bars and/or slop, high or low datapoint weights—see footnote 3 on page 59—and/or other reasons.

¹⁵Minus the “burn-in” of 10,000 initial samples that are discarded, to allow for the Markov Chain to enter the majority of the distribution and not over-sample the outside.

4. The TRK Codebase: Core Algorithms

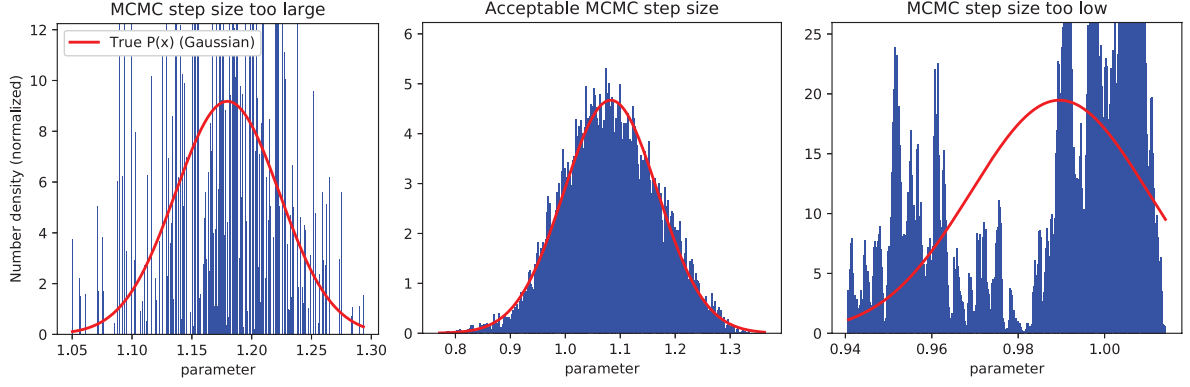


Figure 4.3.: Histograms of MCMC Metropolis-Hastings samplings for a normal-distributed parameter m , with true Gaussian posterior plotted in red, given (Gaussian) proposal distribution widths/“step sizes” of $\sigma_m = 10$, $\sigma_m = 0.1$, and $\sigma_m = 0.0001$, from left to right, given a sample size of $R = 100,000$. Note that even with such a high sample size, the choice of the step size for the model parameter’s Markov Chain will *drastically* change the quality of the sampling.

the right values for the step sizes/proposal covariance matrix of an MCMC sampler is essential; “step size” values that are too large or too small can lead to terrible samplings¹⁶. However, it is very difficult to judge the quality of a choice of proposal distribution step sizes without running the full sampling itself. Because the TRK likelihood \mathcal{L}^{TRK} requires running the tangent point-finding logic for all N data-points at every evaluation of \mathcal{L}^{TRK} , and I have found that it usually takes $R \sim 100,000$ to get a decent sampling of the posterior, $2 \times N \times 100,000$ calls to the tangent-point finding routine are required in order to sample the posterior distribution of the model and slop parameters of a given model and dataset¹⁷. Even with parallelization and other optimizations, this requires a fair amount of computational work, especially if there are many datapoints and/or a complicated multi-dimensional model. As such, simply using trial-and-error to determine the best proposal distribution covariance matrix by iteratively running full samplings is computationally impractical, *especially* if there are many model parameters, and therefore many step sizes that need tuning.

Because of this problem, I wished to efficiently automate the choosing of the proposal distribution covariance matrix, both to need as little user intervention and to require

¹⁶The technical reasons for the erroneous nature of such samplings are beyond the scope of this work, but it is related to the ratio of the number of potential samples accepted by the Metropolis-Hastings criterion to the number of total samples attempted. Too high a step size leads to too many samples being accepted, while too low a step size leads to too few. Initially, I attempted to optimize the $M + 2$ step sizes of the model and slop parameters with respect to this acceptance ratio, but this proved faulty for a number of reasons.

¹⁷The factor of 2 here is due to the Metropolis-Hastings acceptance ratio requiring two calls to the posterior/likelihood function

4. The TRK Codebase: Core Algorithms

the lowest overall computation time as possible. To do so, I used the Adaptive MCMC algorithm of Haario, Saksman, Tamminen, et al. (2001), which updates the (mean and covariance matrix of the) proposal distribution Q while the sampling process is underway, according to the quality of the accumulating total sample. My implementation of the complete Adaptive MCMC sampler is shown as Algorithm 6.

With the Metropolis Hastings and Adaptive MCMC methods, the posterior probability distribution(s) for the model and slop parameters of a model function can be generated, given some dataset and fitting scale. Given these distributions, we can now estimate the (Gaussian) uncertainties/standard deviations of the model parameters, using the following method.

4.4.2. Computing Uncertainties from Parameter Distributions

In order to estimate model parameter uncertainties/error bars given MCMC-sampled posterior distributions, the TRK suite uses what I refer to as the “Bar-Lowering” method from Trotter (2011). For some parameter, a 1σ , 2σ or 3σ confidence interval corresponds to the range of possible parameter values that make up 68.27%, 95.45% and 99.73% of the total integrated probability distribution, respectively. Given a model parameter sample distribution, this method works by first binning the samples into a histogram, and then iteratively finding exactly where 1σ , 2σ and 3σ , i.e. 68.27%, 95.45% and 99.73% of the data lies. This algorithm is explicitly given as Algorithm 7.

Due to the binning nature of this algorithm, it also automatically computes the *asymmetric* confidence intervals/standard deviations of parameters, i.e. $\pm 1\sigma$, $\pm 2\sigma$ and $\pm 3\sigma$ intervals. For example, if the distribution of a parameter is a perfectly symmetric Gaussian, then the -1σ width is equivalent to the $+1\sigma$ width, the -2σ width is equivalent to the $+2\sigma$ width, etc. However, if the distribution is an *asymmetric* Gaussian, e.g. Equation (5.8) (see §5.2 for an in-depth discussion of such asymmetric distributions), all six different widths will be computed.

Algorithm 6: Adaptive Metropolis-Hasting MCMC algorithm for sampling the posterior distribution of model and slop parameters.

```

1 Function OptimizeMCMCProposalDist
    Input : Optimum fitting scale  $s = s_0$  (or any scale of choice), sample size  $R$ 
              (100,000 by default), “burn-in” count  $B$  (10,000 by default), initial
              guess for  $\mathcal{M}$  model and slop parameters/starting point for Markov
              chain  $\Theta_g \equiv \{\vartheta_m, (\sigma_x, \sigma_y)\}_{\text{guess}}$ , dataset  $D \equiv \{x_n, y_n, \sigma_{x,n}, \sigma_{y,n}\}$ , and
              any parameter priors  $p(\Theta)$ .
    Output:  $\{R \text{ samples of } \Theta \text{ from posterior, i.e. } \Theta \sim P(\Theta|D)\}$ 
2 begin
3     Notation:  $\Sigma, \mu$  are the covariance matrix and mean vector of the proposal
       distribution  $Q$ , respectively, which is Gaussian by default.
4     Initialize guess  $\Sigma_i$  for  $\Sigma$  and  $\mu_i = \Theta_g$  for  $\mu$ .
5     Note: we use an automate the guessing of  $\Sigma$ , that can be fairly imprecise;
       however, this algorithm is very guess-independent, and we have never had
       issues with non-rapid convergence.
6     Initialize vector of all accepted samples as  $S = \{\Theta_g\}$ 
7     Initialize Markov Chain sampler starting point  $\Theta_i = \Theta_g$  and define
        $\lambda = 2.38^2/\mathcal{M}$ 
8     while Sample count  $\leq R + B$  do
9         At iteration  $i + 1$ : Sample trial  $\Theta_t$  from posterior  $P(\Theta) = \mathcal{L}^{\text{TRK}}(\Theta)p(\Theta)$ 
           given current proposal  $Q$  with Metropolis-Hastings acceptance
           criterion:
10        while  $\Theta_t$  not accepted do
11             $\Theta_t \sim Q(\mu_i, \lambda\Sigma_i)$  (Gaussian by default)
12             $\alpha = \frac{P(\Theta_t)}{P(\Theta_i)} = \frac{\mathcal{L}^{\text{TRK}}(\Theta_t)p(\Theta_t)}{\mathcal{L}^{\text{TRK}}(\Theta_i)p(\Theta_i)}$  (See footnote 14.)
13            Accept  $\Theta_t$  with probability  $\min(1, \alpha)$ 
14        end
15         $\Theta_{i+1} = \Theta_t$ 
16        Append  $\Theta_{i+1}$  to  $S$ 
17        Update proposal  $Q$ :
18         $\gamma_{i+1} = 1/(i + 1)$ 
19         $\mu_{i+1} = \mu_i + \gamma_{i+1}(\Theta_{i+1} - \mu_i)$ 
20         $\Sigma_{i+1} = \Sigma_i + \gamma_{i+1}[(\Theta_{i+1} - \mu_i)(\Theta_{i+1} - \mu_i)^T - \Sigma_i]$ 
21    end
22    Remove first  $B$  samples from  $S$  (to account for burn-in)
23    return  $S$ 
24 end
    
```

Algorithm 7:

```

1 Function GetUncertainties
   Input : MCMC-sampled model parameter distribution  $S \equiv \{\{\vartheta_m\}\}$  and
           best fit parameters  $\{\vartheta_m, (\sigma_x, \sigma_y)\}$ .
   Output:  $\pm 1\sigma$ -,  $\pm 2\sigma$ - and  $\pm 3\sigma$  confidence intervals/uncertainties of model
           parameters  $\{\vartheta_m\}$ .

2 begin
3   for each model parameter  $\vartheta$  do
4     Make histogram of  $\vartheta$  data from  $S$ , with  $k$  bins total
5     for all  $n\sigma$  of  $\pm 1\sigma$ -,  $\pm 2\sigma$ - and  $\pm 3\sigma$  do
6       Initialize brackets  $h = \max(\text{bins in histogram}), l = 0$ 
7       Initialize bar  $b = h/2$ 
8       Let  $r = (\text{amount of histogram bins} \geq b)/k$ 
9       while  $|r - n\sigma| \geq \text{tolerance}$  do
10        Update  $r$ 
11        if  $r < n\sigma$  then
12           $h = b$ 
13        end
14        else if  $r \geq n\sigma$  then
15           $l = b$ 
16        end
17         $b = (l + h)/2$ 
18      end
19      Let  $-n\sigma =$  midpoint parameter value of leftmost bin above bar
20      Let  $+n\sigma =$  midpoint parameter value of rightmost bin above bar
21    end
22  end
23  return  $\{\pm 1\sigma, \pm 2\sigma, \pm 3\sigma \text{ error bars } \forall \text{ model parameters } \{\vartheta_m\}\}$ 
24 end

```

4.4.3. Reality Check: A Linear TRK Example Fit

At this point I have discussed all of the mechanisms for the basic functionality of the TRK algorithm suite. Before I delve into the advanced/additional topics of the following chapter, I will present a “reality check” of the TRK suite in the form of a basic linear fit. Consider the dataset shown in red in the top left of Figure 4.4¹⁸, and model of the functional form $y = mx + b$, i.e. model parameters of b and m . Running a full TRK fit (i.e. scale optimization, likelihood maximization and then MCMC-generated uncertainty computation) resulted in the model distribution shown in the top left of Figure 4.4. As expected, this slop-dominated dataset (i.e. the small error bars are not enough to account for the scatter of the data) resulted in a linear fit with high slop parameters (σ_x, σ_y) , as evidenced by the fairly large 1-, 2- and 3σ confidence intervals of the model distribution shown¹⁹. After running this fit, I also show the results of the MCMC-generated model parameter distributions (the bottom of Figure 4.4), including a confidence ellipse in parameter space with 1-, 2- and 3σ regions shaded (top right of Figure 4.4), generated via kernel density estimation.

I have now covered all of the essential parts of the TRK fitting suite, and provided an example of its usage. Before I discuss the applications and more complicated examples of TRK fitting, and compare the method to similar algorithms, I will first examine further algorithms and options of the TRK suite, including an automated algorithm for the minimization of model parameter correlation for certain models, and allowing for asymmetric error bars and/or slop parameters.

¹⁸The error bars are possibly too small to see; I choose these error bars to illustrate the fitting behavior for a slop-dominated dataset.

¹⁹These intervals are computed with the slop values and the model parameters, given some model. Specifically, the $n\sigma$ vertical offset of the interval boundaries are given by $\pm n\sqrt{\sigma_y^2 + \left(\frac{dy_c(x)}{dx}\sigma_x\right)^2}$, from Trotter (2011).

4. The TRK Codebase: Core Algorithms

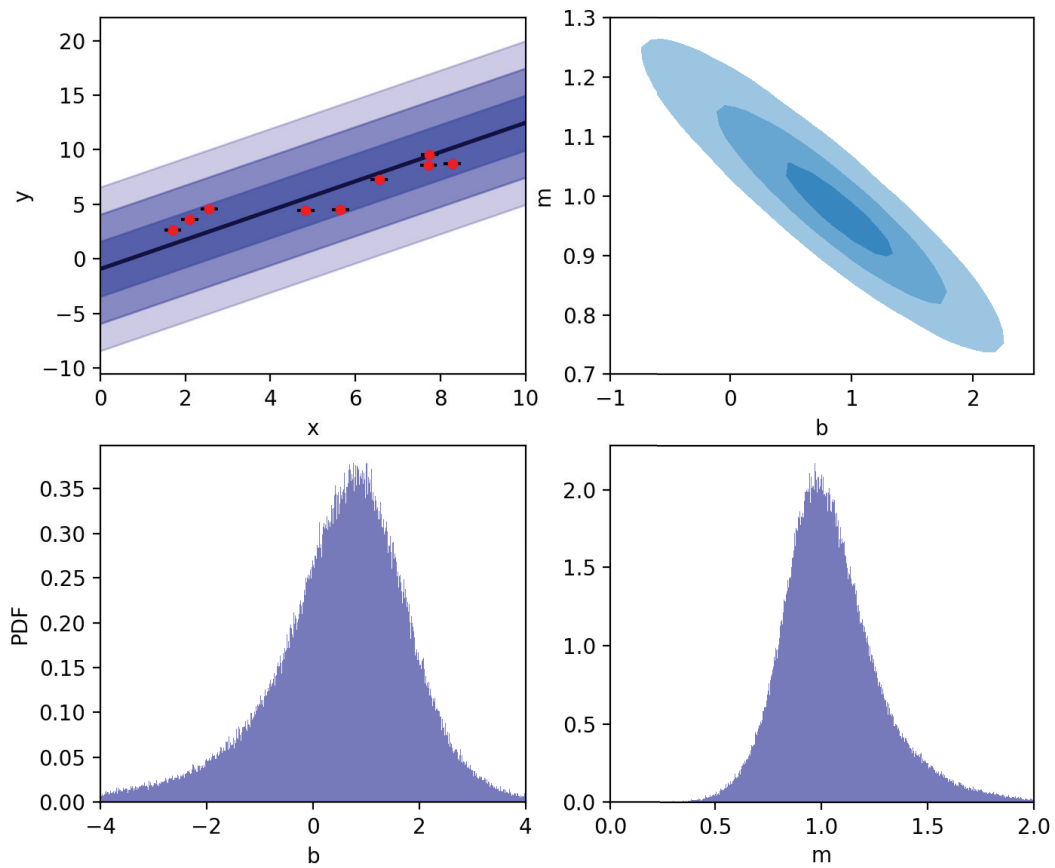


Figure 4.4.: Top left: Example extrinsic scatter-dominated dataset (red) with small x - and y - error bars (barely visible), alongside linear fit model distribution $y = mx + b$ with 1-, 2- and 3σ slope confidence regions visible in blue. Top right: model parameter confidence ellipse with 1-, 2- and 3σ regions shaded, generated from the posterior probability distributions for b and m found via MCMC, bottom.

5. The TRK Codebase: Additional Algorithms

In the previous chapter I examined the basic usage of the TRK statistic to fit models to data, including fitting scale optimization, likelihood maximization, and model parameter probability distribution generation. These features are satisfactory for many fitting tasks, but more complicated situations may arise that require special, generalized treatment. In this chapter, I will first introduce an algorithm developed to remove correlation between model parameters. From here, I will then examine the more complicated, but general case of having asymmetric uncertainties, intrinsic and/or extrinsic, in a dataset, and how this can be fitted to as part of the TRK suite.

5.1. Automatic Model Parameter Correlation Removal and Pivot Points

To begin, consider some linear model of the form $y = mx + b$; typically, the fitted slope m will be correlated with the fitted intercept b , i.e. the confidence ellipse between the two parameters in parameter space will be correlated/tilted. To see why, consider redefining such linear models as $y = m(x - x_p) + b$, with some “pivot point” x_p . Then, the intercept parameter is effectively $b - mx_p$, so given some fitted m , the fitted intercept will be impacted by m , depending on the choice of x_p . Therefore, the uncertainty in b depends on the uncertainty in m . As shown in Trotter (2011), an optimal choice for x_p exists such that the correlation of b and m is minimized (for some dataset).

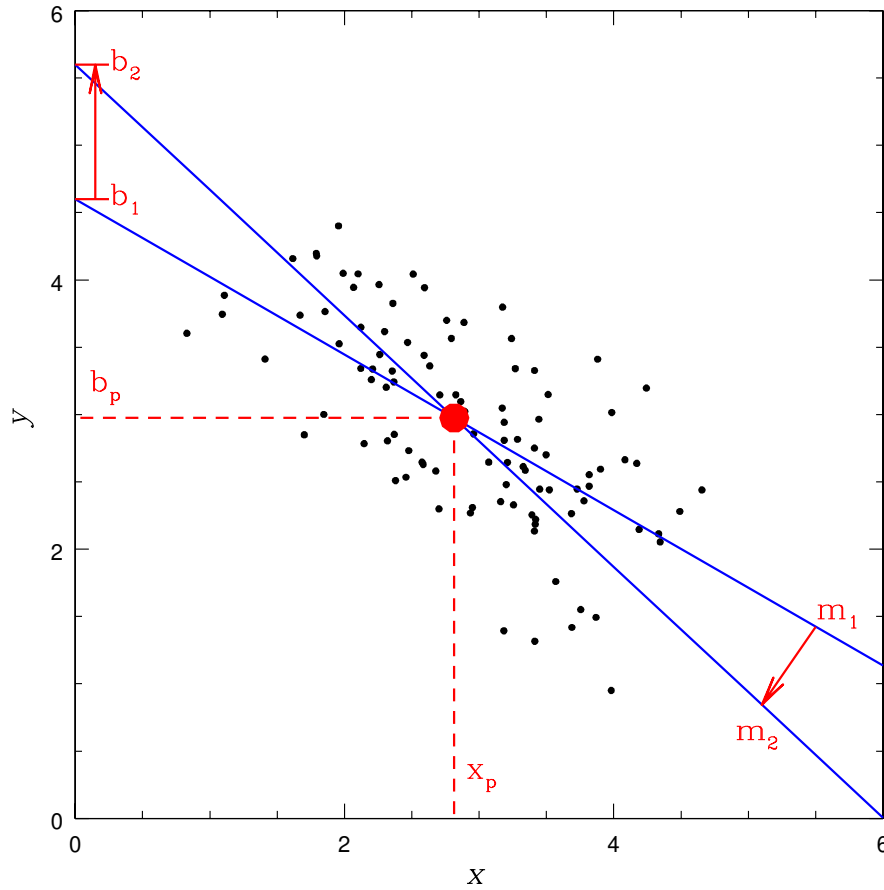


Figure 5.1.: From Trotter (2011), this figure illustrates two different linear fits on the same dataset, with both sets of model parameters b, m offset from some best fit b, m , representing the uncertainty in this best fit. As shown, both of these lines intersect at some optimum pivot point x_p , plotted in red; if this optimal x_p is chosen when fitting the model, the correlation between b and m will be minimized.

5. The TRK Codebase: Additional Algorithms

I have devised an algorithm for automatically determining the pivot point for some model; I note that it can be used not just on linear models, but also any linearizable model, as long as the model can be (re-)written in the form of $y = m(x - x_p) + b$. As an example, consider the power-law model $y(x) = a_0 \left(\frac{x}{10^{x_p}} \right)^{a_1}$. To linearize this, begin by taking the log of both sides as

$$\begin{aligned} \log_{10} y(x) &= \log_{10} \left[a_0 \left(\frac{x}{10^{x_p}} \right)^{a_1} \right] = \log_{10} a_0 + \log_{10} \left(\frac{x}{10^{x_p}} \right)^{a_1} = \log_{10} a_0 + a_1 \log_{10} \left(\frac{x}{10^{x_p}} \right) \\ &= \log_{10} a_0 + a_1 [\log_{10} x - \log_{10} 10^{x_p}] \equiv \log_{10} a_0 + a_1 [\log_{10} x - (\log_{10} x)_p]. \end{aligned} \quad (5.1)$$

As such, the linearized version of the power law has intercept $\log_{10} a_0 \rightarrow b$, slope $a_1 \rightarrow m$, pivot point $(\log_{10} x)_p$, and the data transforms as $\log_{10} y \rightarrow y$ and $\log_{10} x \rightarrow x$.

To introduce the correlation removal/pivot point-finding algorithm, begin by considering two linear (or linearized) fits

$$y = m_1(x - x_p) + b_1 \quad \text{and} \quad y = m_2(x - x_p) + b_2 \quad (5.2)$$

with some shared pivot point x_p (see Figure 5.1). To determine the *optimum* pivot point, we take an iterative approach. As shown in Figure 5.1, the optimum pivot point should be the x -value where the two lines intersect; as such, we set Equations (5.2) equal to each other and solve to obtain

$$x = x_p + \frac{b_1 - b_2}{m_2 - m_1}. \quad (5.3)$$

In order to use Equation (5.3) to solve for the optimum pivot point, we need two sets of slope and intercept parameters. However, what parameters should be used? Because no choice for these parameters is known *a priori*, I take a Monte Carlo-based approach by using the Metropolis Hastings algorithm (Algorithm 6) to generate a large sampling (e.g. $R \sim 10,000$) of intercept and slope parameters (b, m) , given some previous iteration (or guess, if on the first iteration) of the pivot point $x_p = x_p^{\text{old}}$. I then generate $K \sim 100,000$ randomly-drawn 2-combinations of pairs of (b, m) from this sample (i.e. various $\{(b_1, m_1), (b_2, m_2)\}$ of

5. The TRK Codebase: Additional Algorithms

Equation (5.3))¹. From this, I compute 100,000 possible new pivot points x_p^{new} using Equation (5.3). From here, how can we use this *distribution* of x_p^{new} to determine the optimal x_p^{new} , i.e. the next iteration of the true optimal pivot point that will minimize correlation?

To start, I desired to choose a weight $w_{x_p^{\text{new}}}$ for each value of x_p^{new} according to the uncertainty $\sigma_{x_p^{\text{new}}}$ of it's computation, i.e. $w_{x_p^{\text{new}}} = 1/\sigma_{x_p^{\text{new}}}^2$, given the $\{(b_1, m_1), (b_2, m_2)\}$ that were used to compute it². To do so, I used standard propagation of uncertainty to first linear order with Equation (5.3) to obtain

$$\sigma_{x_p^{\text{new}}} \simeq \sqrt{2 \left(\frac{\sigma_b}{m_2 - m_1} \right)^2 + 2 \left[\frac{b_1 - b_2}{(m_2 - m_1)^2} \right]^2 \sigma_m^2}, \quad (5.4)$$

where I have assumed that the uncertainties $\sigma_{b_1} = \sigma_{b_2} \equiv \sigma_b$ and $\sigma_{m_1} = \sigma_{m_2} \equiv \sigma_m$. Next, note that for a linear model $y = m(x - x_p) + b \equiv mx' + b$, we have that $\sigma_b^2 = \sigma_m^2 \overline{(x')^2} = \sigma_m^2 \overline{(x - x_p)^2}$, where the overline indicates an average over all N datapoints (Morrison (2014)). Combining this with Equation (5.4), we now have that

$$\sigma_{x_p^{\text{new}}} \simeq \sqrt{2 \frac{\sigma_m^2 \overline{(x - x_p)^2}}{(m_2 - m_1)^2} + 2 \frac{(b_1 - b_2)^2}{(m_2 - m_1)^4} \sigma_m^2} = \sqrt{2} \sigma_m \sqrt{\frac{\overline{(x - x_p)^2}}{(m_2 - m_1)^2} + \frac{(b_1 - b_2)^2}{(m_2 - m_1)^4}}. \quad (5.5)$$

Given that each x_p^{new} can be weighted according to $w_{x_p^{\text{new}}} = 1/\sigma_{x_p^{\text{new}}}^2$, we can factor out constants of proportionality in Equation (5.5) to obtain

$$w_{x_p^{\text{new}}} \propto \left[\frac{\overline{(x - x_p)^2}}{(m_2 - m_1)^2} + \frac{(b_1 - b_2)^2}{(m_2 - m_1)^4} \right]^{-1}. \quad (5.6)$$

Finally, note that for our iterative approach, x ranges over the sample of the $K = 100,000$ new pivot points x_p^{new} computed from Equation (5.3), and x_p is the previous iteration's optimal single pivot point x_p^{old} . As such, a single i^{th} new pivot point $x_{p,i}^{\text{new}}$ computed using

¹I note that although I don't use explicit mechanisms to avoid duplicate 2-combinations, there are $\binom{10,000}{2} \simeq 5 \times 10^7$ possible randomly-chosen 2-combinations from the set of 10,000 samples, so if 100,000 are combinations drawn, there is only a $\simeq 0.2\%$ chance of having a single duplicate.

²This relationship between weight and uncertainty assumes (approximately) Gaussian error.

5. The TRK Codebase: Additional Algorithms

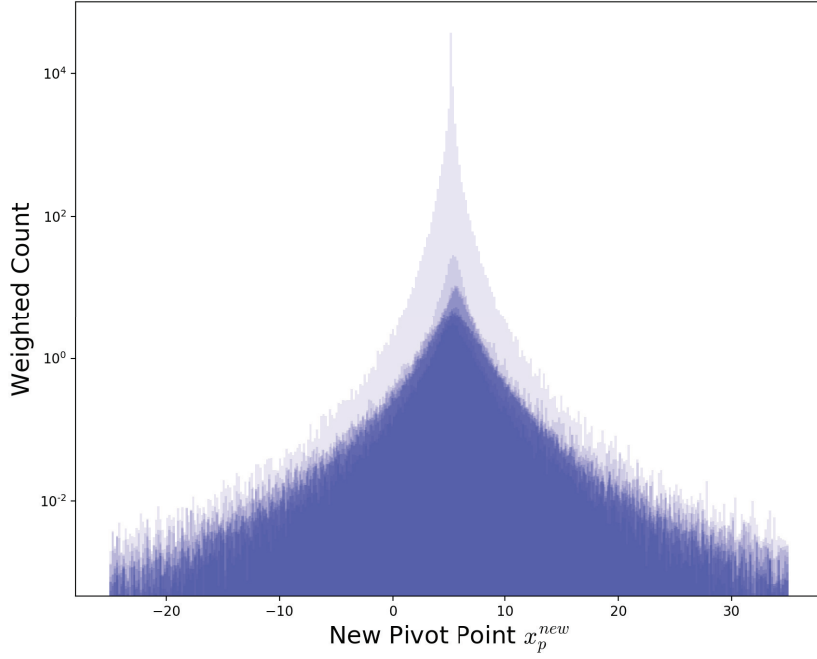


Figure 5.2.: Ten different iterations for the MCMC-generated distributions of new pivot points x_p^{new} computed with Equation (5.3) and weighted according to Equation (5.7), overlapped. For each iteration we take the optimal value of the new pivot point to be the weighted half-sample mode (see Bickel and Fruehwirth (2005)) of that iteration's distribution, and use this optimal value as x_p^{old} for the next iteration in order to compute the next distribution of x_p^{new} . As can be seen (given that only four or so iterations are visible), I have found this algorithm to converge quickly, usually only needing a few iterations, if more than one is needed at all.

some $\{(b_1, m_1), (b_2, m_2)\}$ with Equation (5.3) is weighted according to

$$\begin{aligned}
 w_{x_p^{\text{new}}, i} &\propto \left[\frac{\sum_{j=1}^K \left[\left(x_p^{\text{old}} + \frac{b_{1,j} - b_{2,j}}{m_{2,j} - m_{1,j}} \right) - x_p^{\text{old}} \right]^2}{K(m_2 - m_1)^2} + \frac{(b_1 - b_2)^2}{(m_2 - m_1)^4} \right]^{-1} \\
 &= \left[\frac{\sum_{j=1}^K \left(\frac{b_{1,j} - b_{2,j}}{m_{2,j} - m_{1,j}} \right)^2}{K(m_2 - m_1)^2} + \frac{(b_1 - b_2)^2}{(m_2 - m_1)^4} \right]^{-1}, \tag{5.7}
 \end{aligned}$$

which is the final expression that is used. Finally, I take the weighted half-sample mode (i.e. Bickel and Fruehwirth (2005)) of the $K = 100,000$ values for x_p^{new} weighted according to Equation (5.7), and use that as the next iteration for the optimum x_p . From here, I iterate until convergence.

I summarize the pivot point-finding/correlation removal method in Algorithm 8. Once the final pivot point has been found, it is stored as part of the model, and the rest of the

5. The TRK Codebase: Additional Algorithms

TRK algorithm can be used as normal. As an example of this method, Fig. 5.3 shows the results of running the pivot point optimization algorithm on the same linear fit dataset as in Figure 4.4 of §4.4.3, then running the regular TRK scale optimization, fit and MCMC uncertainty computation as usual. Observe in this figure how the fitted slope m is mostly unchanged from Figure 4.4, but the intercept b is noticeably different. Most notably, the parameter space confidence ellipse of b and m is no longer tilted, indicating a loss of correlation between the two parameters. Quantifiably, note that for the fit in Figure 4.4 where the pivot point was not optimized beforehand, the sampled (b, m) has a Pearson correlation coefficient of $R^2 = -0.502$, indicating (negative) correlation as expected. On the other hand, when the pivot point *is* optimized before running this fit, shown in Fig. 5.3, the same correlation coefficient is only $R^2 = 0.056 \simeq 0$, indicating almost no correlation, as expected.

The ability to minimize parameter correlation for linearized models is a useful and widely applicable addition to the TRK suite. Next, in order to further generalize the TRK statistic, I will explore the possibility of introducing asymmetric error bars and/or slop parameters.

5.2. Asymmetric Intrinsic and/or Extrinsic Uncertainty

Throughout this thesis I have only considered models that have intrinsic and extrinsic scatter (i.e. error bars and slop, respectively) that follow *symmetric* Gaussian distributions, i.e. Equation (2.4). However, there remains the distinct possibility for either or both the error bars and the slop/extrinsic scatter to follow *asymmetric* distributions, such as the skew-normal distribution. A number of examples of asymmetric error bars within datasets and/or asymmetric slop within models are given in Trotter (2011), and as shown in that work, a reasonable approximation of such distributions is defined as the (normalized) *asymmetric* Gaussian distribution,

$$\mathcal{N}_A(x'; \mu, \sigma_+, \sigma_-) \equiv \frac{2}{\sqrt{2\pi}(\sigma_+ + \sigma_-)} \begin{cases} \exp\left\{-\frac{1}{2}\left(\frac{x' - \mu}{\sigma_+}\right)^2\right\} & \text{if } x' \geq \mu \\ \exp\left\{-\frac{1}{2}\left(\frac{x' - \mu}{\sigma_-}\right)^2\right\} & \text{if } x' < \mu \end{cases} . \quad (5.8)$$

Algorithm 8: Method to remove model parameter correlation/determine pivot point for linearized models

```

1 Function FindPivotPoint
    Input : Linearized model of the form  $y_c(x) = m(x - x_p) + b$ , dataset
              $D \equiv \{x_n, y_n, \sigma_{x,n}, \sigma_{y,n}\}$ , and initial guess for optimum pivot point
              $x_p^{\text{old}}$ .
    Output: Optimum pivot point  $x_p$  that minimizes correlation between model
             parameters  $b$  and  $m$ .

2 begin
3     Initialize  $R = 10,000, K = 100,000$  (by default)
4     while  $x_p^{\text{new}}$  not converged do
5         Initialize empty arrays  $\mathcal{X}, \mathcal{W}$  for new pivot points and their weights,
            respectively
6         Generate  $R$  samples of  $(b, m) \sim \text{MetHastSampler}()$  (see Algorithm 6)
7         Store these samples as the set  $\mathcal{P}$ 
8         Compute  $K$  new pivot points:
9         for  $i = 1, 2, \dots, K$  do
10            Randomly draw pair  $\{(b_1, m_1), (b_2, m_2)\}$ . from  $\mathcal{P}$ 
11            Let  $x_{p,i}^{\text{new}} = x_p^{\text{old}} + \frac{b_1 - b_2}{m_2 - m_1}$ 
12            Store  $x_{p,i}^{\text{new}}$  in  $\mathcal{X}$ 
13        end
14        Weight these  $K$  new pivot points:
15        for  $i = 1, 2, \dots, K$  do
16            Let weight  $w_{x_{p,i}^{\text{new}}} = \left[ \frac{\sum_{j=1}^K \left( \frac{b_{1,j} - b_{2,j}}{m_{2,j} - m_{1,j}} \right)^2}{K(m_2 - m_1)^2} + \frac{(b_1 - b_2)^2}{(m_2 - m_1)^4} \right]^{-1}$ 
17            Store  $w_{x_{p,i}^{\text{new}}}$  in  $\mathcal{W}$ 
18        end
19        Compute optimum  $x_p^{\text{new}} = \text{WeightedHalfSampleMode}(\mathcal{X}, \mathcal{W})$ 
20         $x_p^{\text{new}} \rightarrow x_p^{\text{old}}$ 
21    end
22 end
23 return  $x_p = x_p^{\text{new}}$ 
    
```

5. The TRK Codebase: Additional Algorithms

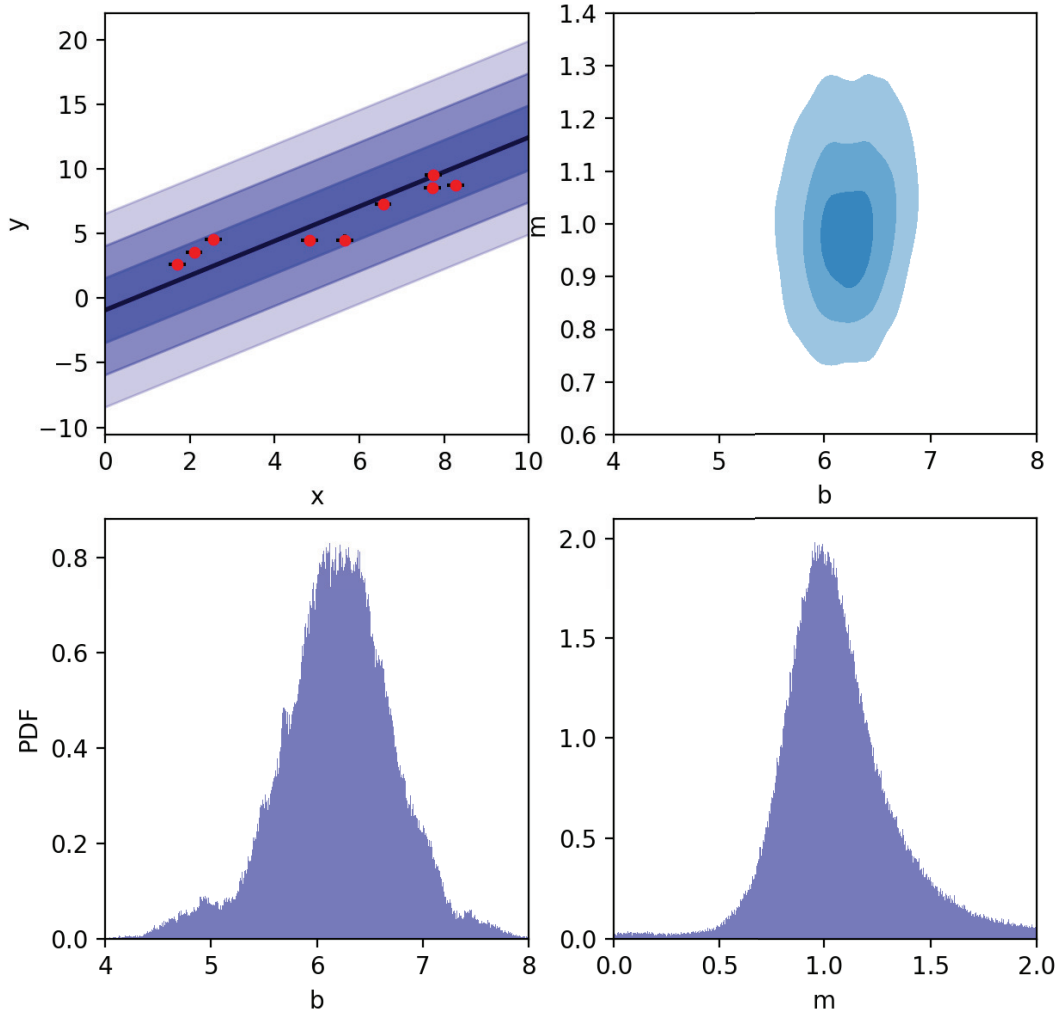


Figure 5.3.: Top left: The same extrinsic scatter-dominated dataset as in Figure 4.4 but now with pivot point x_p in model $y = m(x - x_p) + b$ optimized to minimize correlation between b and m , using Algorithm 8. Model is shown with 1-, 2- and 3 σ slope confidence regions visible in blue. Top right: model parameter confidence ellipse with 1-, 2- and 3 σ regions shaded, generated from the posterior probability distributions for b and m found via MCMC, bottom. Note that with pivot point optimized, the confidence ellipse is now no longer tilted as in Fig. 4.4, indicating the removal of correlation, also evidenced by a Pearson correlation coefficient of $R^2 = 0.056 \simeq 0$.

5. The TRK Codebase: Additional Algorithms

From here, a 2D asymmetric Gaussian is simply the product of two of these 1D asymmetric Gaussians. Note that this is just a general case of the typical symmetric Gaussian, as this expression simplifies to Equation (2.4) when $\sigma_- = \sigma_+ \equiv \sigma$. In the next section, following the work of Trotter (2011), I will show how this distribution can be introduced to the TRK statistic in the form of both asymmetric error bars and/or slop.

5.2.1. Theory

I will first introduce some notation. Following Trotter (2011), I parameterize the asymmetric x and y error bars of some n^{th} datapoint with some parameters $\{\sigma_{x,n+}, \sigma_{x,n-}\}$ and $\{\sigma_{y,n+}, \sigma_{y,n-}\}$, respectively, following Equation (5.8). If asymmetric slop/extrinsic scatter along x and/or y is also modeled, such asymmetry is characterized with parameters $\{\sigma_{x+}, \sigma_{x-}\}$ and $\{\sigma_{y+}, \sigma_{y-}\}$, respectively. Now, in order to perform TRK fits on asymmetric data and/or models, because I have fundamentally generalized the concept of intrinsic and extrinsic scatter, the TRK statistic must be re-developed following the analysis of §2 with this new asymmetric formalism. To begin, consider a model distribution about the model curve $y_c(x; \vartheta_m)$ that has asymmetric Gaussian slops $(\sigma_{x\pm}, \sigma_{y\pm})$, alongside a single n^{th} datapoint with asymmetric Gaussian error bars $(\sigma_{x,n\pm}, \sigma_{y,n\pm})$, illustrated in Figure 5.4. Delving into this section, observe that the joint probability of this data-point p_n requires evaluating the convolution of the two asymmetric Gaussians defined by $(\sigma_{x\pm}, \sigma_{y\pm})$ and $(\sigma_{x,n\pm}, \sigma_{y,n\pm})$, according to Equation 2.6.

As shown in Trotter (2011), the convolution of two asymmetric Gaussian distributions does not itself exactly result in an asymmetric Gaussian. However, Trotter showed that an excellent approximation of such a convolution is a 2D asymmetric Gaussian with widths defined by $(\Sigma_{x,n\pm}, \Sigma_{y,n\pm})$ and centroid defined by $(x_n + \delta_{x,n}, y_n + \delta_{y,n})$, where

$$\begin{aligned}\Sigma_{x,n\pm} &\equiv (\sigma_{x\mp}^2 + \sigma_{x,n\pm}^2)^{1/2} \\ \Sigma_{y,n\pm} &\equiv (\sigma_{y\mp}^2 + \sigma_{y,n\pm}^2)^{1/2}\end{aligned}\tag{5.9}$$

and $(\delta_{x,n}, \delta_{y,n})$ describes an offset from the data-point which can be found using an empirical method developed by Trotter, given in Algorithm 9, of which the derivation of is beyond the scope of this work. Given that the likelihood is the product of all N of the

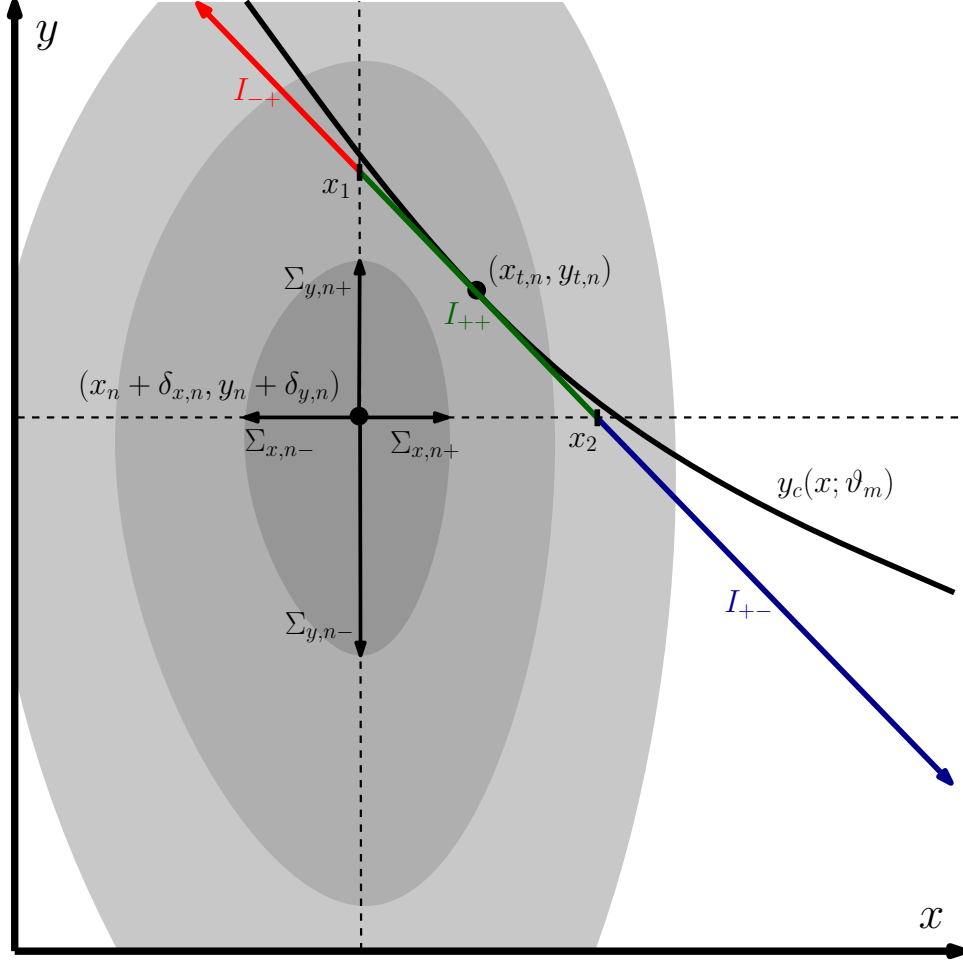


Figure 5.4.: From Trotter (2011), a visualization of the 2D joint probability for an asymmetric model curve y_c and data-point (x_n, y_n) with asymmetric error bars. This joint probability is an integral that can be approximated by a 2D asymmetric Gaussian with widths $(\Sigma_{x,n\pm}, \Sigma_{y,n\pm})$ centered at some $(x_n + \delta_{x,n}, y_n + \delta_{y,n})$. The integral is broken into three segments according to the quadrants about the convolved centroid $(x_n + \delta_{x,n}, y_n + \delta_{y,n})$, in this case I_{-+} , I_{++} and I_{+-} , respectively, with subscripts denoting which quadrant the integral/linear approximation of the model curve is located in. For this example, I_{-+} (red) through quadrant 2 has limits $(-\infty, x_1]$, I_{++} (green) through quadrant 1 has limits $[x_1, x_2]$, and I_{+-} (blue) through quadrant 4 has limits $[x_2, \infty)$.

5. The TRK Codebase: Additional Algorithms

joint probabilities of the datapoints, using this 2D asymmetric Gaussian approximation for the integral in Equation (2.6) used to compute p_n gives rise to an interesting observation. From Trotter (2011), in this asymmetric case the likelihood will have the form of a χ^2 -like statistic, but with the datapoint shifted from (x_n, y_n) by $(\delta_{x,n}, \delta_{y,n})$, with $\pm 1\sigma$ error bars expanded in quadrature by the \pm -components of the model slop.

In order to compute the joint probability p_n for a single datapoint in this asymmetric case, I will first refer back to the example given in Figure 5.4 and follow §B of Trotter (2011). As mentioned, we can approximate the convolution of the two 2D asymmetric Gaussians from Equation (2.6) as a single, shifted 2D asymmetric Gaussian with widths $(\Sigma_{x,n\pm}, \Sigma_{y,n\pm})$ and centroid $(x_n + \delta_{x,n}, y_n + \delta_{y,n})$. Then, following the analysis of §2.2, I approximate the model curve $y_c(x; \vartheta_m)$ as a line tangent to the *asymmetric* convolved (and shifted) error “ellipse” of the datapoint, such that solving Equation (2.13) for the tangent point $x_{t,n}$ requires changing $(\Sigma_{x,n}, \Sigma_{y,n})$ according to which quadrant about the ellipse that the tangent point lies in. For example, if the tangent point is in Quadrant I as in Fig. 5.4, Equation (2.13) transforms with $(\Sigma_{x,n}, \Sigma_{y,n}) \rightarrow (\Sigma_{x,n+}, \Sigma_{y,n+})$.

Observe Equation (2.15), the symmetric version of the approximate analytic expression for p_n , and note the factor $\frac{du_n}{dx}$. This factor is dependent on the choice of rotated coordinate system (u_n, v_n) , following §2.2.3. Combining the choice of (u_n, v_n) that creates the TRK statistic and this asymmetric case, I then find that $\frac{du_n}{dx}$ becomes

$$\frac{du_n}{dx} \equiv \left(\frac{du_n}{dx} \right)_{\pm\mp} = \frac{m_{t,n}^2 \Sigma_{x,n\pm}^2 + \Sigma_{y,n\mp}^2}{\sqrt{m_{t,n}^2 \Sigma_{x,n\pm}^4 + \Sigma_{y,n\mp}^4}}, \quad (5.14)$$

where the subscripts $(\pm\mp)$ are determined by which quadrant the tangent point lines in, i.e $\left(\frac{du_n}{dx}\right)_{++}$ corresponds to quadrant I, $\left(\frac{du_n}{dx}\right)_{-+}$ to quadrant II, $\left(\frac{du_n}{dx}\right)_{--}$ to quadrant III, and $\left(\frac{du_n}{dx}\right)_{+-}$ to quadrant IV. In other words, the first subscript indicates the choice of $\Sigma_{x,n\pm}$ and the second indicates the choice of $\Sigma_{y,n\mp}$ in Equation (5.14).

Proceeding on, I note that in order to obtain an analytic expression for an asymmetric p_n analogous to Equation (2.15), we must evaluate the $(-\infty, \infty)$ integral of Equation (2.14). In general, this joint probability integral must now be broken up into three separate integrals (I_1, I_2, I_3) , as the tangent line can cross through up to three quadrants of the

Algorithm 9: Method from Trotter (2011) used to determine offsets $(\delta_{x,n}, \delta_{y,n})$ from data-point (x_n, y_n) that are used to compute the 2D asymmetric Gaussian joint probability distribution for some n^{th} data-point.

1 **Function** *GetAsymmShifts*

Input : Asymmetric slop parameters $(\sigma_{x\pm}, \sigma_{y\pm})$ and asymmetric error bars $(\sigma_{x,n\pm}, \sigma_{y,n\pm})$

Output : Offset $(\delta_{x,n}, \delta_{y,n})$

2 **begin**

3 Initialize $\sigma_L \equiv \max\{\sigma_{y+}, \sigma_{y-}\}$, $\sigma_S \equiv \min\{\sigma_{y+}, \sigma_{y-}\}$,
 4 $\sigma_{n,L} \equiv \max\{\sigma_{y,n+}, \sigma_{y,n-}\}$, $\sigma_{n,S} \equiv \min\{\sigma_{y,n+}, \sigma_{y,n-}\}$, and
 5 $\sigma_{\max} \equiv \max\{\sigma_L, \sigma_{n,L}\}$.

6 Let $\xi \equiv \frac{\sigma_S}{\sigma_L} + \frac{\sigma_{n,S}}{\sigma_{n,L}}$ and $\eta \equiv \begin{cases} \frac{\sigma_{n,S}}{\sigma_{n,L}} - \frac{\sigma_S}{\sigma_L} & \text{if } \sigma_{n,L} < \sigma_L \\ \frac{\sigma_S}{\sigma_L} - \frac{\sigma_{n,S}}{\sigma_{n,L}} & \text{if } \sigma_{n,L} \geq \sigma_L \end{cases}$.

7 Define $r \equiv \frac{\min\{\sigma_L, \sigma_{n,L}\}}{\max\{\sigma_L, \sigma_{n,L}\}}$, $\xi' = \begin{cases} \xi & \text{if } \xi \leq 1 \\ 2 - \xi & \text{if } \xi > 1 \end{cases}$ and

8 $\eta' = \begin{cases} 0 & \text{if } \xi' = 0 \\ 2\xi' \left[\frac{1}{2} \frac{\eta}{\xi'} + 1 \right]^{n(r)} - \xi' & \text{otherwise} \end{cases}$, where $n(r) \equiv r^{-0.4087}$.

9 Compute $\delta_* = \sigma_{\max} N(r) [f(\xi)g(\eta') + h(\xi)]$, given N, f, g and h :

10

$$N(r) = -0.5326r^2 + 1.5307r + 0.0019 \quad (5.10)$$

$$f(\xi) = \begin{cases} 0 & \text{if } \xi = 0, \\ 0.2454\xi^{-1.1452} & \text{if } \xi \leq 1, \\ 0.2454\xi^{-0.5203} & \text{if } \xi > 1, \end{cases} \quad (5.11)$$

$$g(\eta') = \eta'^2 \quad (5.12)$$

$$h(\xi) = -0.042\xi^2 - 0.1602\xi + 0.4884. \quad (5.13)$$

if *One of the distributions is symmetric* **then**

11 Let $i = \begin{cases} +1 & \text{if } \sigma_{n,L} = \sigma_{y,n+} \text{ or } \sigma_L = \sigma_{y-} \\ -1 & \text{if } \sigma_{\max} = \sigma_{y,n-} \text{ or } \sigma_{\max} = \sigma_{y+} \end{cases}$
 12 $\delta_{y,n} = i \times \delta_*$

13 **end**

14 **if** *Both distributions are asymmetric* **then**

15 Let $i = \begin{cases} +1 & \text{if } \sigma_{\max} = \sigma_{y,n+} \text{ or } \sigma_{\max} = \sigma_{y-} \\ -1 & \text{if } \sigma_{\max} = \sigma_{y,n-} \text{ or } \sigma_{\max} = \sigma_{y+} \end{cases}$

16 **if** $\sigma_L = \sigma_{y-}$ and $\sigma_{n,L} = \sigma_{y,n+}$, or $\sigma_L = \sigma_{y+}$ and $\sigma_{n,L} = \sigma_{y,n-}$ **then**

17 | $\delta_{y,n} = i \times \delta_*$

18 **end**

19 **else**

20 | $\delta_{y,n} = i \times \delta_* \times \sin\left(\frac{\pi \eta'}{2 \xi'}\right) \times \begin{cases} \xi^{0.7413} & \text{if } \xi \leq 1 \\ \xi^{-0.1268} & \text{if } \xi > 1 \end{cases}$

21 **end**

22 **end**

23 Repeat the above but with $y \rightarrow x$ to compute $\delta_{x,n}$.

24 **return** $(\delta_{x,n}, \delta_{y,n})$

25 **end**

5. The TRK Codebase: Additional Algorithms

ellipse, such that $p_n = I_1 + I_2 + I_3$. Following Figure 5.4, these three integrals will have limits of $(-\infty, x_1]$, $[x_1, x_2]$ and $[x_2, \infty)$, where $x_1 = x_n + \delta_{x,n}$ and x_2 is the point where the tangent line intersects with the shifted datapoint centroid, i.e. $y_{t,n} + m_{t,n}(x - x_{t,n}) = y_n + \delta_{y,n}$. Note that for all three integrals, the factor $(\frac{du_n}{dx})_{\pm\mp}$ is the same, given that it only depends on which quadrant the tangent point lies in. Notating the Gaussian cumulative distribution function as

$$\Phi(z) \equiv \int_{-\infty}^z e^{-\frac{1}{2}x^2} dx, \quad (5.15)$$

evaluating these three integrals with the asymmetric distributions results in the first integral being

$$\begin{aligned} I_1 = & \left(\frac{du_n}{dx} \right) \frac{\kappa \Sigma_{x,n\pm} \Sigma_{y,n\mp} \Phi(z_{\pm\mp}(x_1))}{\sqrt{m_{t,n}^2 \Sigma_{x,n\pm}^2 + \Sigma_{y,n\mp}^2}} \\ & \times \exp \left\{ -\frac{1}{2} \left[\frac{y_{t,n} - y_n - \delta_{y,n} - m_{t,n}(x_{t,n} - x_n - \delta_{x,n})}{\sqrt{m_{t,n}^2 \Sigma_{x,n\pm}^2 + \Sigma_{y,n\mp}^2}} \right]^2 \right\} \end{aligned} \quad (5.16)$$

with normalization factor³ $\kappa \equiv 2/[\pi(\Sigma_{x,n+} + \Sigma_{x,n-})(\Sigma_{y,n+} + \Sigma_{y,n-})]$, where the two sign subscripts of $(\Sigma_{x,n\pm}, \Sigma_{y,n\mp})$, and of the transformed limit of integration

$$z_{\pm\mp}(x) = \frac{\Sigma_{y,n\mp}^2(x - x_n - \delta_{x,n}) + m_{t,n}^2 \Sigma_{x,n\pm}^2 [x - x_{t,n} - (y_n + \delta_{y,n} - y_{t,n})/m_{t,n}]}{\Sigma_{x,n\pm} \Sigma_{y,n\mp} \sqrt{m_{t,n}^2 \Sigma_{x,n\pm}^2 + \Sigma_{y,n\mp}^2}} \quad (5.17)$$

are dependent on which two $\Sigma_{x,n\pm}, \Sigma_{y,n\mp}$ widths of the ellipse bound the quadrant that this first integral goes through. For example, in the case of Figure 5.4, the tangent point is in quadrant I, so this first integral is in quadrant II, which is bounded by $\Sigma_{x,n-}$ and $\Sigma_{y,n+}$, so in this case Equation (5.16) simplifies to

$$\begin{aligned} I_1 = & \left(\frac{du_n}{dx} \right) \frac{\kappa \Sigma_{x,n-} \Sigma_{y,n+} \Phi(z_{\pm\mp}(x_1))}{\sqrt{m_{t,n}^2 \Sigma_{x,n-}^2 + \Sigma_{y,n+}^2}} \\ & \times \exp \left\{ -\frac{1}{2} \left[\frac{y_{t,n} - y_n - \delta_{y,n} - m_{t,n}(x_{t,n} - x_n - \delta_{x,n})}{\sqrt{m_{t,n}^2 \Sigma_{x,n-}^2 + \Sigma_{y,n+}^2}} \right]^2 \right\}. \end{aligned} \quad (5.18)$$

³Note that this normalization factor is the same for all three integrals within any of the four quadrants of the ellipse.

5. The TRK Codebase: Additional Algorithms

The second integral is similarly found to be

$$\begin{aligned}
 I_2 = & \left(\frac{du_n}{dx} \right) \frac{\kappa \Sigma_{x,n\pm} \Sigma_{y,n\mp} [\Phi(z_{\pm\mp}(x_2)) - \Phi(z_{\pm\mp}(x_1))]}{\sqrt{m_{t,n}^2 \Sigma_{x,n\pm}^2 + \Sigma_{y,n\mp}^2}} \\
 & \times \exp \left\{ -\frac{1}{2} \left[\frac{y_{t,n} - y_n - \delta_{y,n} - m_{t,n}(x_{t,n} - x_n - \delta_{x,n})}{\sqrt{m_{t,n}^2 \Sigma_{x,n\pm}^2 + \Sigma_{y,n\mp}^2}} \right]^2 \right\}, \quad (5.19)
 \end{aligned}$$

and the third is found to be

$$\begin{aligned}
 I_3 = & \left(\frac{du_n}{dx} \right) \frac{\Sigma_{x,n\pm} \Sigma_{y,n\mp} [1 - \Phi(z_{\pm\mp}(x_2))]}{\sqrt{m_{t,n}^2 \Sigma_{x,n\pm}^2 + \Sigma_{y,n\mp}^2}} \\
 & \times \exp \left\{ -\frac{1}{2} \left[\frac{y_{t,n} - y_n - \delta_{y,n} - m_{t,n}(x_{t,n} - x_n - \delta_{x,n})}{\sqrt{m_{t,n}^2 \Sigma_{x,n\pm}^2 + \Sigma_{y,n\mp}^2}} \right]^2 \right\}, \quad (5.20)
 \end{aligned}$$

where again, the sign subscripts of $(\Sigma_{x,n\pm}, \Sigma_{y,n\mp})$ and $z_{\pm\mp}(x)$ are dependent on which two $\Sigma_{x,n\pm}, \Sigma_{y,n\mp}$ widths of the ellipse bound the quadrant that the respective integral goes through. Then, continuing with the example in Figure 5.4, the second integral is in Quadrant I while the third is in Quadrant IV, so we have that

$$\begin{aligned}
 I_2 = & \left(\frac{du_n}{dx} \right) \frac{\kappa \Sigma_{x,n+} \Sigma_{y,n+} [\Phi(z_{++}(x_2)) - \Phi(z_{++}(x_1))]}{\sqrt{m_{t,n}^2 \Sigma_{x,n+}^2 + \Sigma_{y,n+}^2}} \\
 & \times \exp \left\{ -\frac{1}{2} \left[\frac{y_{t,n} - y_n - \delta_{y,n} - m_{t,n}(x_{t,n} - x_n - \delta_{x,n})}{\sqrt{m_{t,n}^2 \Sigma_{x,n+}^2 + \Sigma_{y,n+}^2}} \right]^2 \right\}, \quad (5.21)
 \end{aligned}$$

and

$$\begin{aligned}
 I_3 = & \left(\frac{du_n}{dx} \right) \frac{\Sigma_{x,n+} \Sigma_{y,n-} [1 - \Phi(z_{+-}(x_2))]}{\sqrt{m_{t,n}^2 \Sigma_{x,n+}^2 + \Sigma_{y,n-}^2}} \\
 & \times \exp \left\{ -\frac{1}{2} \left[\frac{y_{t,n} - y_n - \delta_{y,n} - m_{t,n}(x_{t,n} - x_n - \delta_{x,n})}{\sqrt{m_{t,n}^2 \Sigma_{x,n+}^2 + \Sigma_{y,n-}^2}} \right]^2 \right\}. \quad (5.22)
 \end{aligned}$$

From here, the joint probability of the datapoint is found with $p_n = I_1 + I_2 + I_3$, which is

5. The TRK Codebase: Additional Algorithms

used to define the TRK statistic (i.e. the likelihood) in the same way as the symmetric case.

5.2.2. Implementation

The introduction of asymmetric error bars and/or slop to the TRK suite manifests in a number of ways, the first being the change in the TRK likelihood function \mathcal{L}^{TRK} . I have shown how to compute the joint probability p_n for a single n^{th} data-point with asymmetric error bars and/or slop, by adding up the three integrals I_1 , I_2 and I_3 of Equations (5.16), (5.19) and (5.20), respectively. From here, \mathcal{L}^{TRK} is simply the product of all N of the p_n of the datapoints, following Equation (2.7). An essential, but not necessarily obvious thing to point out is that now, there are (up to) *four* slop parameters in total, $(\sigma_{x+}, \sigma_{x-}, \sigma_{y+}, \sigma_{y-}) \equiv (\sigma_{x\pm}, \sigma_{y\pm})$. As such, maximizing \mathcal{L}^{TRK} / minimizing $-2 \ln \mathcal{L}^{\text{TRK}}$ to find a best fit with the Nelder-Mead method as described in §4.2 will now require performing the optimization with respect to all $M + 4$ parameters, given M non-slop model parameters. Furthermore, as discussed on page 51, the driving equation of the tangent point-finding routine, Equation (2.13), must be evaluated with the correct two $(\Sigma_{x,n\pm}, \Sigma_{y,n\pm})$, according to which quadrant of the datapoint ellipse the tangent point is located in. I also note that the *shifted* data-point centroid $(x_n + \delta_{x,n}, y_n + \delta_{y,n})$, found with Algorithm 9, must be used in place of the unshifted (x_n, y_n) of the symmetric case at all steps of running a TRK fit. Finally, I note that the scale optimization procedure of §4.3 does not obviously generalize for the asymmetric case; this issue will be tackled in the future, as discussed in §7.

6. Applications and Examples

The final major chapter of this work will first compare the TRK statistic/suite to similar methods in §6.1, and schematically demonstrate the improvements over thereof of TRK. From here, I will demonstrate the TRK fitting algorithms in practice within §6.2, by performing TRK fits with models that involve parameters related to the dust-extinction of interstellar light.

6.1. Comparison to Similar Algorithms

Surprisingly, there are not very many algorithms that can be used to fit models to data that has uncertainty in two dimensions (even fewer that can account for both intrinsic and extrinsic uncertainties). To further examine the usefulness of the TRK suite, I will begin by comparing it to various similar algorithms, and showing how, overall, it is usually the most robust and general choice. First, I will discuss a non-Bayesian least-squares algorithm, then we will explore two Bayesian algorithms that are similar, but inferior to TRK.

Perhaps the most well known/most-used 2D-uncertainty fitting method is Orthogonal Distance Regression/ODR, also known as Total Least Squares¹. ODR is a nonlinear least-squares regression method that minimizes the distances between the datapoints and the fitted curve along some direction(s) determined by the error bars of the data, e.g. Brown, Fuller, et al. (1990). Despite being heavily used, there are a number of downsides to this method as compared to the TRK statistic, for example:

¹For example, one of the most commonly used implementations of ODR is the `scipy.odr` Python module (see <https://docs.scipy.org/doc/scipy/reference/odr.html>.)

6. Applications and Examples

1. There is no obvious method to include/parameterize extrinsic scatter/slop in the dataset with ODR.
2. There is no general method to incorporate ODR into a Bayesian formalism, e.g. including priors, that we have found.
3. As shown by Isobe et al. (1990), ODR is *not* scalable/scale invariant, i.e. changing the units of the axis/axes will result in different, unequivalent fits.

As such, when a rigorous, flexible and generalizable fitting algorithm is desired over pure computational speed, TRK will be a better choice over ODR.

As described in §2.2.3, the choice of the arbitrary rotated coordinates (u_n, v_n) and therefore the factor $\frac{du_n}{dx}$ in Equation (2.16)—the likelihood function in our general case—will be what defines a given statistic’s criteria for a best fit. Trotter (2011) showed that various choices of $\frac{du_n}{dx}$ will lead to different statistics with varying properties. There exist two other 2D-uncertainty Bayesian model-fitting statistics in the literature, both of which are thoroughly explored and compared to the TRK statistic in Trotter’s work: that of D’Agostini (2005), or D05, and that of Daniel E Reichart (2001), or R01. In the following, we will summarize Trotter’s work of showing how both of these statistics are derived from certain choices of $\frac{du_n}{dx}$, and comparing them to the TRK statistic.

The D05 statistic of D’Agostini (2005) is defined by setting $du_n = dx$, i.e. $\frac{du_n}{dx} = 1$ in Equation (2.16), giving a likelihood function of

$$\begin{aligned} \mathcal{L}^{\text{D05}} &\propto \prod_{n=1}^N \frac{1}{\sqrt{m_{t,n}^2 \Sigma_{x,n}^2 + \Sigma_{y,n}^2}} \exp \left\{ -\frac{1}{2} \frac{[y_n - y_{t,n} - m_{t,n}(x_n - x_{t,n})]^2}{m_{t,n}^2 \Sigma_{x,n}^2 + \Sigma_{y,n}^2} \right\} \\ -2 \ln \mathcal{L}^{\text{D05}} &= \sum_{n=1}^N \frac{[y_n - y_{t,n} - m_{t,n}(x_n - x_{t,n})]^2}{m_{t,n}^2 \Sigma_{x,n}^2 + \Sigma_{y,n}^2} + \sum_{n=1}^N \ln(m_{t,n}^2 \Sigma_{x,n}^2 + \Sigma_{y,n}^2) + C. \end{aligned} \quad (6.1)$$

From Trotter (2011), “The D05 statistic can be seen to be analogous to a one-dimensional χ^2 statistic in y , where the difference between the model and the datapoint is the difference between the tangent line at $x = x_n$ and y_n , and where the 1σ uncertainty in the convolved datapoint is replaced by the quadrature sum of $\Sigma_{y,n}$ and $\Sigma_{x,n}$ projected into the y -direction using the slope $m_{t,n}$. D05 differs from a traditional χ^2 statistic in that the denominator of the argument of the exponential, and the prefactor of the exponential are themselves functions of the slopes (σ_x, σ_y) , which are treated as free model parameters.” However, as

6. Applications and Examples

shown by Trotter, while D05 is scalable and reduces to a 1D χ^2 -like statistic in the limit of $\Sigma_{x,n} \rightarrow 0$, it is *not* invertible, unlike the TRK statistic.

The R01 statistic of Daniel E Reichart (2001) is defined using $du_n = ds_n$, where $ds_n \equiv \sqrt{dx^2 + dy^2}$ is parallel to the tangent line with slope $m_{t,n}$, giving $\frac{du_n}{dx} = \sqrt{1 + m_{t,n}^2}$. Using Equation (2.16), this results in a likelihood of

$$\begin{aligned} \mathcal{L}^{\text{R01}} &\propto \prod_{n=1}^N \sqrt{\frac{1 + m_{t,n}^2}{m_{t,n}^2 \Sigma_{x,n}^2 + \Sigma_{y,n}^2}} \exp \left\{ \left\{ -\frac{1}{2} \frac{[y_n - y_{t,n} - m_{t,n}(x_n - x_{t,n})]^2}{m_{t,n}^2 \Sigma_{x,n}^2 + \Sigma_{y,n}^2} \right\} \right\} \\ -2 \ln \mathcal{L}^{\text{R01}} &= \sum_{n=1}^N \frac{[y_n - y_{t,n} - m_{t,n}(x_n - x_{t,n})]^2}{m_{t,n}^2 \Sigma_{x,n}^2 + \Sigma_{y,n}^2} \\ &\quad - \sum_{n=1}^N \ln \left(\frac{1 + m_{t,n}^2}{m_{t,n}^2 \Sigma_{x,n}^2 + \Sigma_{y,n}^2} \right) + C. \end{aligned} \quad (6.2)$$

The R01 statistic was designed to be invertible (Daniel E Reichart (2001)); however, as shown in Trotter (2011), although it *is* invertible, the statistic is neither scalable, nor reduces to a 1D χ^2 -like statistic in the limit of $\Sigma_{x,n} \rightarrow 0$.

While D05 is scalable and reduces to a 1D χ^2 -like statistic in the limit of $\Sigma_{x,n} \rightarrow 0$, it is not invertible. Similarly, while R01 is invertible, it is not scalable and it does not reduce to a 1D χ^2 -like statistic in the aforementioned limit. The TRK statistic, however, manages to be the best of both worlds, as it is invertible (see §3.0.1), reduces to a 1D χ^2 -like statistic (see §2.2.2 and Trotter (2011)), and by using the scale optimization algorithm of §4.3, can be made to be scale invariant.

6.2. Interstellar Extinction Model Parameter Fits

In Trotter (2011), various TRK fits were made to model correlations between parameters that describe empirical fits to the observed spectral extinction by dust of light originating from stars in the Milky Way and Magellanic Clouds. These fits served to be a “proof of concept” of Trotter’s “science code” implementation of the TRK statistic. Now that I have developed the TRK statistic in the form of a much more automated, generalizable and computationally efficient implementation, I will redo these fits, with updated datasets, as follows.

6. Applications and Examples

The aforementioned dust extinction models were first formalized in Cardelli, Clayton, and Mathis (1989) (CCM) and Edward L Fitzpatrick and Derck Massa (1988) (FM), and are summarized and described in Figure 6.1. In the following section, I will explore the relationships between the empirical parameters c_1 , c_2 , c_3 , and γ of the CCM/FM model. As described in Trotter (2011), there are known correlations between c_1 and c_2 , and between the so-called “bump height” parameter $BH \equiv c_3/\gamma^2$ (see Fig 6.1) and c_2 . The dataset that was fit to in Trotter (2011) is comprised of extinction parameter measurements from 417 stars in the Milky Way, from Valencic, Clayton, and Gordon (2004), and 23 stars from the Large and Small Magellanic Clouds, from Gordon et al. (2003). This dataset contains values and symmetric error bars for c_1 , c_2 , c_3 and γ (as well as other CCM/FM parameters that don’t concern our fits), through which Trotter derived values and error bars for BH through standard propagation of uncertainty².

For this thesis, following a thorough search through the literature, I updated the full dataset with data from 328 stars in the Milky Way from E. Fitzpatrick and Massa (2007), and one star in M31 from Clayton et al. (2015). From here, I redid the fits of the models describing c_1 vs. c_2 and BH vs. c_2 that were formulated and presented in Trotter (2011), with the updated dataset of $N = 729$ values of $\{c_1, c_2, BH\}$ with symmetric error bars (again computing error bars for $BH \equiv c_3/\gamma^2$ with standard propagation of uncertainty). Following Trotter (2011), I assigned some weight w_n to each n^{th} datapoint that is inversely proportional to the integral of the sum of all N of the intrinsic Gaussian c_2 distributions of the datapoints, weighted by the Gaussian of the corresponding $c_{2,n}$, i.e.

$$w_n \propto \left[\int_{-\infty}^{\infty} \left(\mathcal{N}(c_2|c_{2,n}, \sigma_{c_{2,n}}) \sum_{i=1}^N \mathcal{N}(c_2|c_{2,i}, \sigma_{c_{2,i}}) \right) dc_2 \right]^{-1}, \quad (6.3)$$

and then normalizing the weights such that the minimum weight is unity³. This weighting is used in order to increase the weight of/“oversample” the occasional high c_2 datapoint (Trotter (2011)). From here, I ran TRK fits for c_1 vs. c_2 and BH vs. c_2 , that will be

²The parameters from Valencic, Clayton, and Gordon (2004) were calculated with a normalization given the parameter R_V by Trotter (2011), and their error bars were assigned through standard propagation of uncertainty.

³Datapoint weights $\{w_n\}$ are implemented into the TRK likelihood by multiplying the terms within the summations of the lower line of Equation (2.17) by w_n , which translates to a *weighted* likelihood of

$$\mathcal{L}^{\text{TRK}} \propto \prod_{n=1}^N \left(\frac{m_{t,n}^2 \Sigma_{x,n}^2 + \Sigma_{y,n}^2}{m_{t,n}^2 \Sigma_{x,n}^4 + \Sigma_{y,n}^4} \right)^{w_n/2} \times \exp \left\{ -\frac{1}{2} w_n \frac{[y_n - y_{t,n} - m_{t,n}(x_n - x_{t,n})]^2}{m_{t,n}^2 \Sigma_{x,n}^2 + \Sigma_{y,n}^2} \right\}.$$

6. Applications and Examples

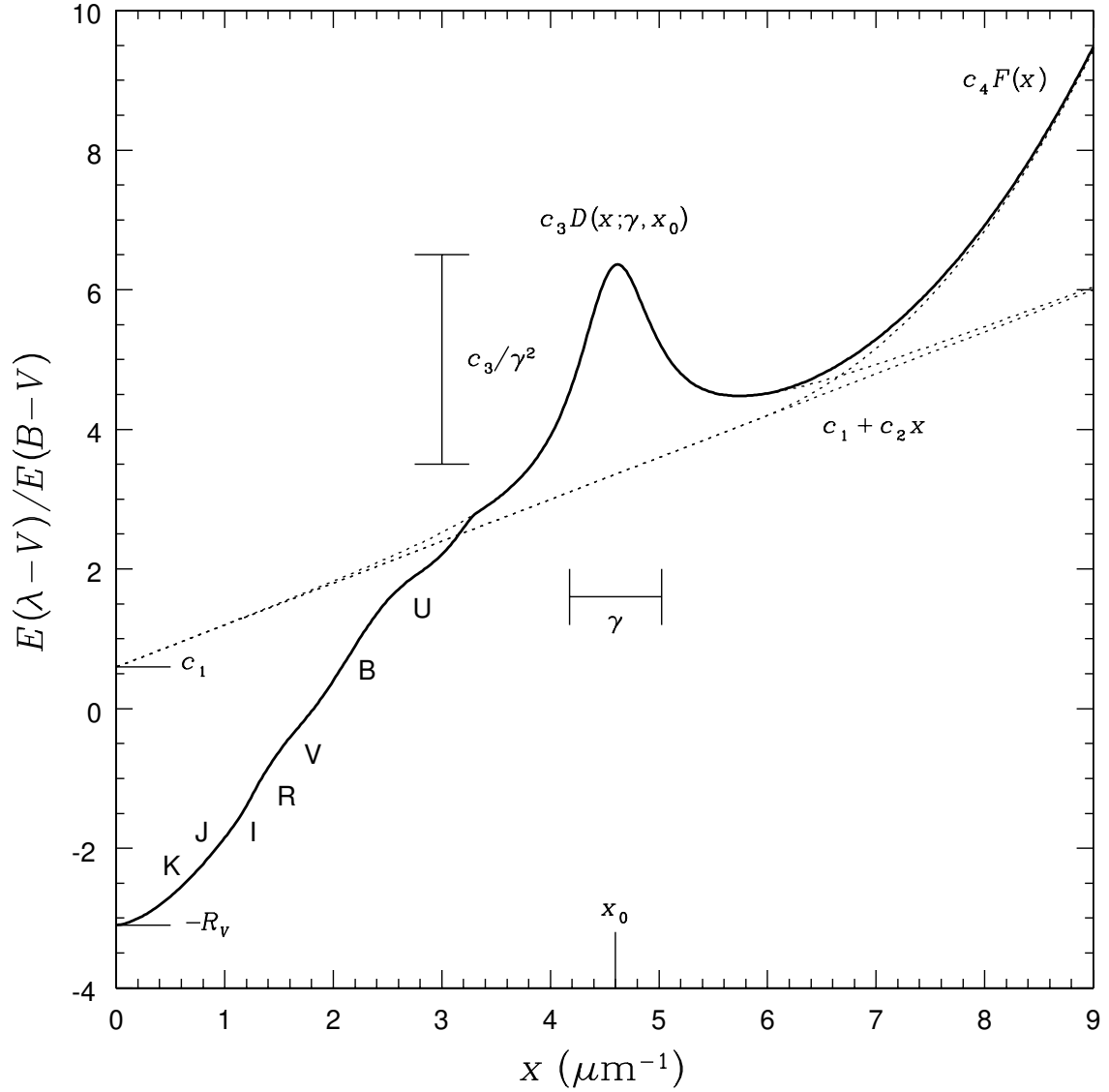


Figure 6.1.: From Trotter (2011), the combined CCM and FM extinction model curve along a typical Milky Way line of sight. The y -axis is defined to be the ratio between the $\lambda - V$ color excess $E(\lambda - V)$ at a given wavelength λ , and the $B - V$ color excess $E(B - V)$, where the color excess is defined as the difference, in magnitudes, of the absorption due to dust at two given wavelengths. In this case, B and V correspond to wavelengths in the middle of standard blue and visible photometric filters. The color excess ratio plotted on the y -axis is thus proportional (with an offset) to the magnitude of dust extinction at a given wavelength λ . The x -axis is proportional to the inverse of λ , specifically $x \equiv (\lambda/1 \mu\text{m})^{-1}$, or, equivalently, proportional to frequency. As shown, the parameters c_1 and c_2 describe the intercept and slope of a linear component of the model. The parameter γ parameterizes the width of the “bump” in the center of the model (also known as the “UV bump”), while the parameter $BH \equiv c_3/\gamma^2$ describes the height of this bump.

6. Applications and Examples

described and presented in the following subsections.

6.2.1. Fitting c_1 vs. c_2

From Trotter (2011), the parameters c_1 and c_2 are strongly linearly correlated, as the two describe the intercept and slope of a linear component of the FM extinction model. The model distribution that I will use to parameterize this correlation, defined by Trotter, has a model curve $y_c(x; \vartheta_m)$ of the form

$$c_{1,c}(c_2; \vartheta_m) = b^{c_1} + m^{c_1} (c_2 - c_2^{p_{c_1}}), \quad (6.4)$$

where $\vartheta_m = \{b^{c_1}, m^{c_1}\}$ are the model parameters and $c_2^{p_{c_1}}$ is the pivot point of the model, of which the choice of affects the amount of correlation between b^{c_1} and m^{c_1} (see §5.1).

To fit this model to the dataset, I first ran the scale optimization algorithm of §4.3 to determine the optimum fitting scale s_0 . Next, I ran the parameter correlation removal/pivot point-finding algorithm of §5.1 to determine the pivot point $c_2^{p_{c_1}}$ that minimizes the correlation between b^{c_1} and m^{c_1} . From here, I determined the best fit model and $(x-, y-)$ slop parameters $\vartheta_m = \{b^{c_1}, m^{c_1}\}$ and $\{\sigma_{c_2}^{c_1}, \sigma_{c_1}\}$, respectively using the Downhill Simplex method of §4.2 at s_0 ⁴. Finally, I used the Adaptive MCMC and Bar-Lowering methods of §4.4.1 to sample the distribution of the model and slop parameters, and compute their uncertainties, respectively.

The numerical results of this fit are given in Table 6.2.1, while the fit is plotted in Figure 6.2 with generated model and slop parameter distributions shown in Figure 6.3. In Figure 6.2, note the strong correlation between c_1 and c_2 , as expected. Also, note that because the pivot point-finding routine was used to minimize the correlation between b^{c_1} and m^{c_1} , the confidence ellipse for these two parameters (top, center of Figure 6.3) is not tilted, indicating the removal of correlation between them. Finally, also note in this figure that the two slop parameters ($\sigma_{c_2}^{c_1}$ and σ_{c_1}) are also uncorrelated, again evidenced by a non-tilted confidence ellipse (bottom, center).

⁴Note that I was able to run the scale optimization before finding the pivot point because fitting scales are invariant of choice of pivot point, as they don't affect the extreme scale limiting behavior of slops, e.g. Trotter (2011).

6. Applications and Examples

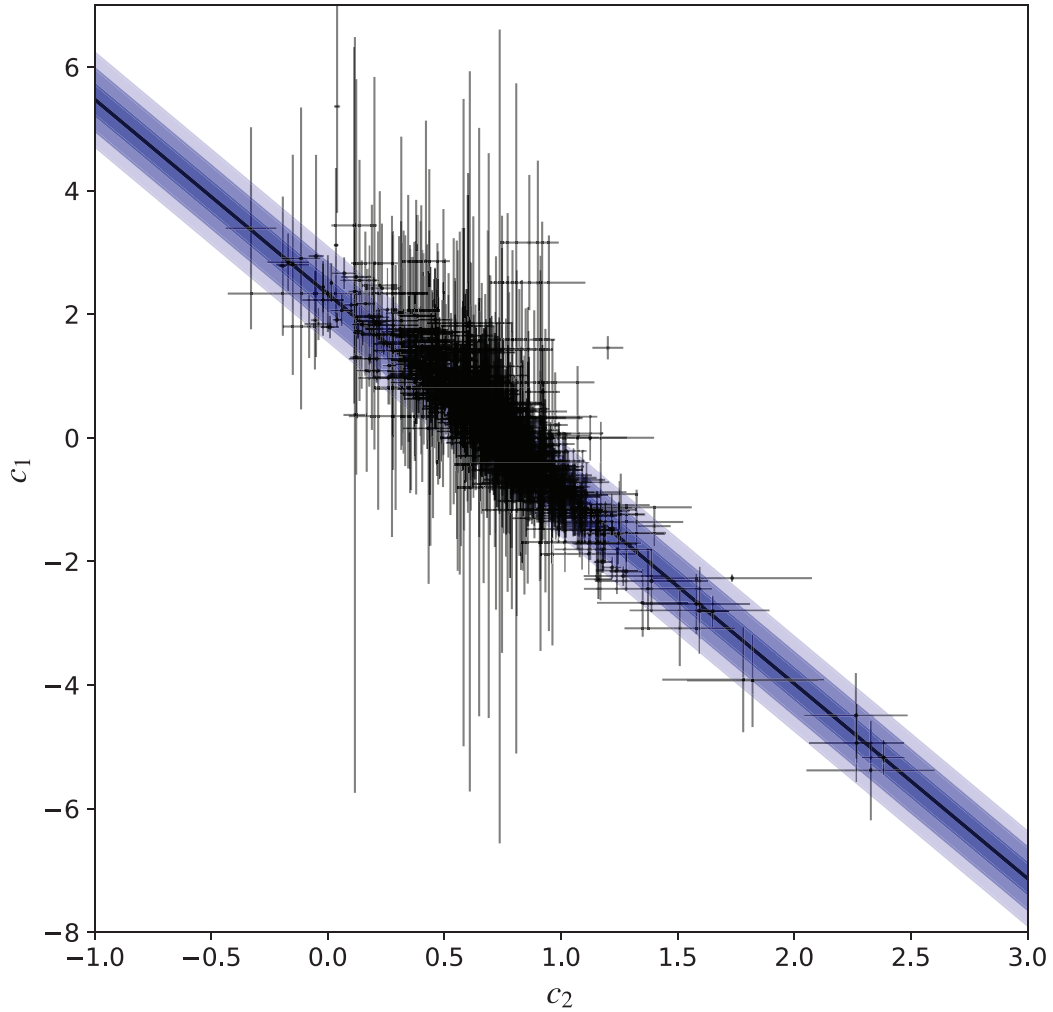


Figure 6.2.: Observed c_1 vs c_2 data from Valencic, Clayton, and Gordon (2004), Gordon et al. (2003), E. Fitzpatrick and Massa (2007), and Clayton et al. (2015), plotted with linear TRK fit modeled by Equation (6.4). Shaded regions indicate the 1-, 2- and 3σ slop confidence regions of the model distribution, given best fit slop values of Table 6.2.1 and plotted according to footnote 19 on page 39.

6. Applications and Examples

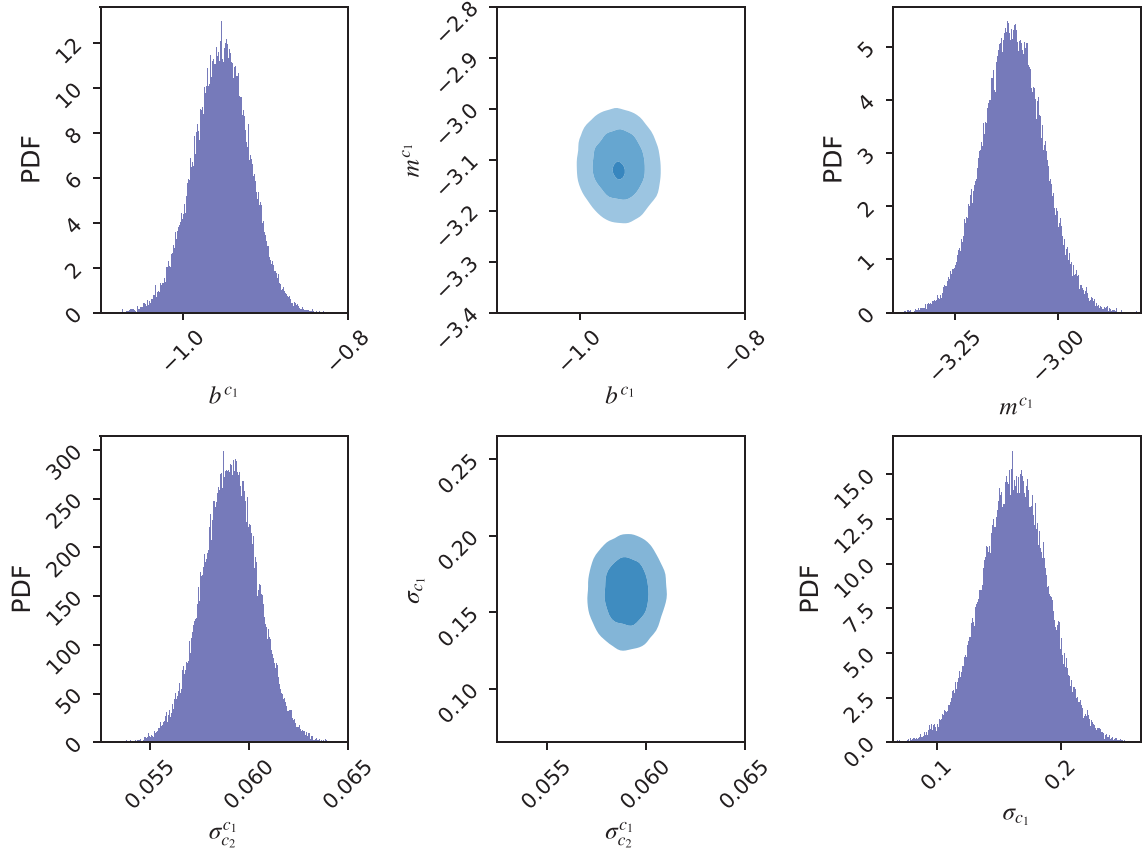


Figure 6.3.: MCMC-generated (§4.4.1) probability distributions for c_1 vs. c_2 (Equation (6.4)) model (top) and slop/extrinsic scatter (bottom) parameters. Parameter confidence ellipses (center) with 1–, 2– and 3 σ regions show joint posterior probabilities with respect to the parameters plotted on either side. The pivot-point finding algorithm of §5.1 was used to remove correlation between model parameters.

Table 6.1.: Best-Fit Asymmetric Gaussian Parameter Values For c_1 vs. c_2 Model (Equation (6.4))

Parameter Type	Parameter	Value	+1 σ Width	–1 σ Width
Model Parameters	b^{c_1}	–0.9052	0.0350	–0.0376
	m^{c_1}	–3.1519	0.0778	–0.0762
Pivot Point	$c_2^{p_{c_2}}$	1.0247
Slop Parameters	$\sigma_{c_2}^{c_1}$	0.05948	0.00135	–0.00148
	σ_{c_1}	0.1711	0.0270	–0.0263
Optimum Scale	s_0	0.28230
Minimum Scale	a	0.15332
Maximum Scale	b	0.44043

6.2.2. Fitting BH vs. c_2

Trotter (2011) also reported that the bump height parameter $\text{BH} \equiv c_3/\gamma^2$ is loosely correlated with c_2 , as moderate values of c_2 tend to have higher values of BH while low or high values of c_2 tend to have low BH, along Milky Way lines-of-sight. Trotter parameterized this relationship with a “smoothly-broken linear” model of the form

$$\text{BH}_c(c_2; \vartheta_m) = -\ln \left[\exp \left\{ -b_1^{\text{BH}} - \tan \theta_1^{\text{BH}} \left(c_2 - c_2^{p_1, \text{BH}} \right) \right\} + \exp \left\{ -b_2^{\text{BH}} - \tan \theta_2^{\text{BH}} \left(c_2 - c_2^{p_2, \text{BH}} \right) \right\} \right], \quad (6.5)$$

where $\vartheta_m = \{b_1^{\text{BH}}, \theta_1^{\text{BH}}, b_2^{\text{BH}}, \theta_2^{\text{BH}}\}$ are the model parameters and $\{c_2^{p_1, \text{BH}}, c_2^{p_2, \text{BH}}\}$ are the pivot points of the model that determine the amount of correlation between b_1^{BH} and θ_1^{BH} , and b_2^{BH} and θ_2^{BH} , respectively⁵.

To fit this model to the dataset, I first ran the scale optimization algorithm of §4.3 to determine the optimum fitting scale s_0 . Next, I ran the parameter correlation removal/pivot point-finding algorithm of §5.1 to determine the pivot points $c_2^{p_1, \text{BH}}$ and $c_2^{p_2, \text{BH}}$ that minimize the correlations between b_1^{BH} and θ_1^{BH} , and b_2^{BH} and θ_2^{BH} , respectively⁶. From here, I determined the best fit model and $(x-, y-)$ slop parameters $\vartheta_m = \{b_1^{\text{BH}}, \theta_1^{\text{BH}}, b_2^{\text{BH}}, \theta_2^{\text{BH}}\}$ and $(\sigma_{c_2}^{\text{BH}}, \sigma_{\text{BH}})$, respectively using the Downhill Simplex method of §4.2 at s_0 . Finally, I used the Adaptive MCMC and Bar-Lowering methods of §4.4.1 to sample the distribution of the model and slop parameters, and compute their uncertainties, respectively. Furthermore, for this fit I used priors on the slope angle parameters θ_1^{BH} and θ_2^{BH} to constrain the angles of the line to specific quadrants (also in order to demonstrate the usage of priors with the

⁵To see the intuition behind this broken-linear parameterization, observe how for low c_2 , the model approximately becomes linear, i.e. $\text{BH}_c(c_2; \vartheta_m) \simeq b_1^{\text{BH}} + \tan \theta_1^{\text{BH}} (c_2 - c_2^{p_1, \text{BH}})$. Similarly, for high c_2 , $\text{BH}_c(c_2; \vartheta_m) \simeq b_2^{\text{BH}} + \tan \theta_2^{\text{BH}} (c_2 - c_2^{p_2, \text{BH}})$. As such, b_1^{BH} and b_2^{BH} are analogous to the intercept parameters for two lines, while θ_1^{BH} and θ_2^{BH} are analogous to (the angles off of the c_2 -axis of) the slope parameters.

⁶Note that the pivot point-finding algorithm of §5.1 and Algorithm 8 was only explicitly defined for models with a single pivot point. As such, I modified the code to allow for models that have multiple pivot points using a fairly straightforward generalization of Algorithm 8. Specifically, I used the MCMC-sampled pairs of slope and intercept parameters (line 6 of Algorithm 8) to compute and weight samples from successive distributions of possible pivot points (lines 8 through 17), in order to iteratively determine optimal values for the two pivot points (lines 19 and 20).

6. Applications and Examples

Table 6.2.: Best-Fit Asymmetric Gaussian Parameter Values For BH vs. c_2 Model (Equation (6.5))

Parameter Type	Parameter	Value	+1 σ Width	−1 σ Width
Model Parameters	b_1^{BH}	1.9837	0.0054	−0.0057
	θ_1^{BH}	4.5679	0.0016	−0.0015
	b_2^{BH}	2.4040	0.0008	−0.0008
	θ_2^{BH}	−1.1344	0.0130	−0.0138
Pivot Points	$c_2^{p_1, \text{BH}}$	−0.0867
	$c_2^{p_2, \text{BH}}$	1.3343
Slop Parameters	$\sigma_{c_2}^{\text{BH}}$	0.1290	0.0037	−0.0039
	σ_{BH}	0.4954	0.0077	−0.0081
Optimum Scale	s_0	0.28117
Minimum Scale	a	0.11816
Maximum Scale	b	0.94043

TRK suite) in the form of

$$p(\theta_1^{\text{BH}}) = \begin{cases} \mathcal{U}(\pi, \frac{3\pi}{2}) & \text{if } \theta_1^{\text{BH}} \in [\pi, \frac{3\pi}{2}], \\ 0 & \text{otherwise} \end{cases}, \quad p(\theta_2^{\text{BH}}) = \begin{cases} \mathcal{U}(-\frac{\pi}{2}, 0) & \text{if } \theta_2^{\text{BH}} \in [-\frac{\pi}{2}, 0], \\ 0 & \text{otherwise,} \end{cases} \quad (6.6)$$

where $\mathcal{U}(a, b)$ indicates a uniform distribution with bounds a and b (note that these priors do not in any way constrain the data itself).

The numerical results of this fit are given in Table 6.2.2, while the fit is plotted in Figure 6.4 with generated model and slop parameter distributions shown in Figure 6.5. Note that because the pivot point-finding routine was used to minimize the correlation between b_1^{BH} and θ_1^{BH} , and b_2^{BH} and θ_2^{BH} , the confidence ellipses for these two pairs of parameters (top, center, and middle, center of Figure 6.5) are not tilted, indicating the removal of correlation between these sets of parameters. Finally, also note in this figure that the two slop parameters ($\sigma_{c_2}^{\text{BH}}$ and σ_{BH}) are also uncorrelated, again evidenced by a non-tilted confidence ellipse (bottom, center).

6.3. A Web-Based TRK Fit Calculator

Alongside with the development of the TRK suite source code, the other main project that I undertook is the end-to-end development of a web-based calculator for running TRK fits,

6. Applications and Examples

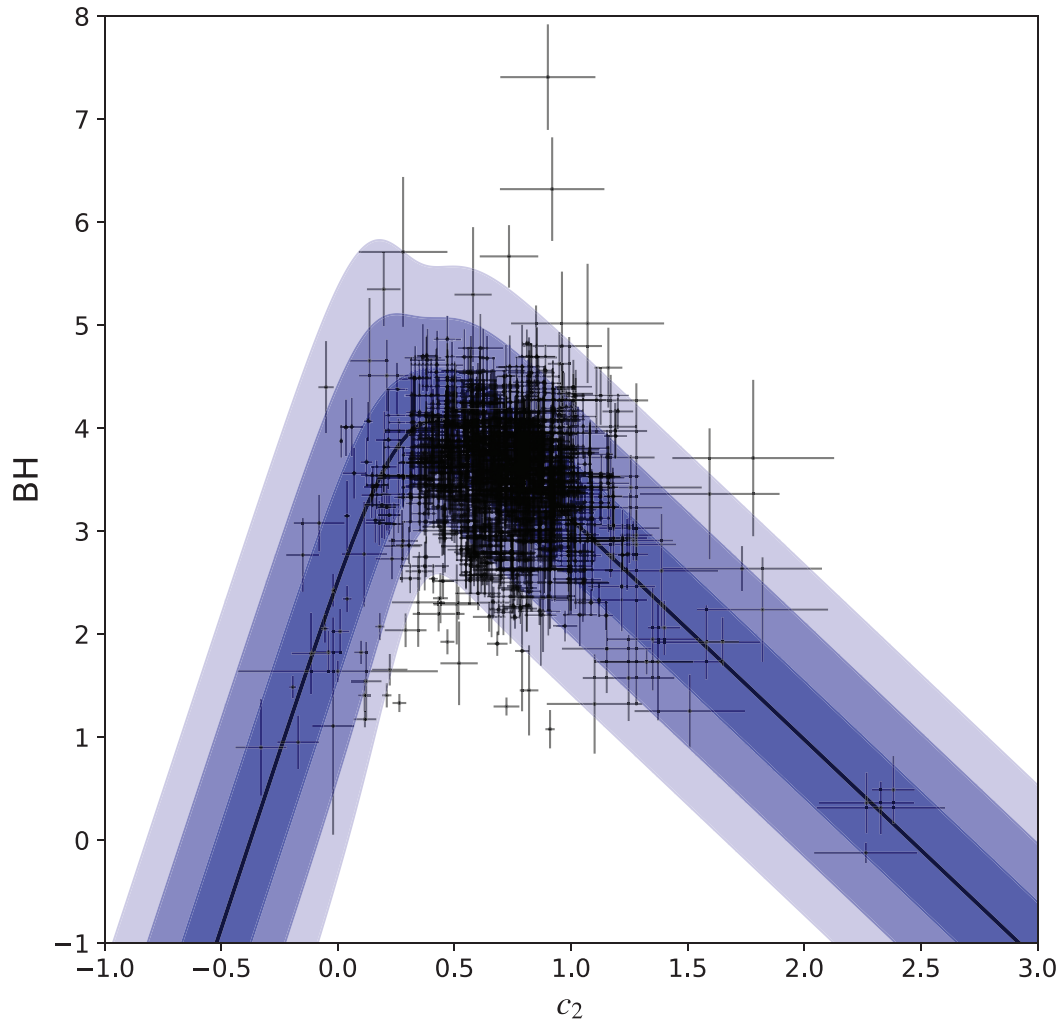


Figure 6.4.: Observed BH vs c_2 data from Valencic, Clayton, and Gordon (2004), Gordon et al. (2003), E. Fitzpatrick and Massa (2007), and Clayton et al. (2015), plotted with broken-linear TRK fit modeled by Equation (6.5). Shaded regions indicate the 1-, 2- and 3σ slope confidence regions of the model distribution, given best fit slope values of Table 6.2.2 and plotted according to footnote 19 on page 39.

6. Applications and Examples

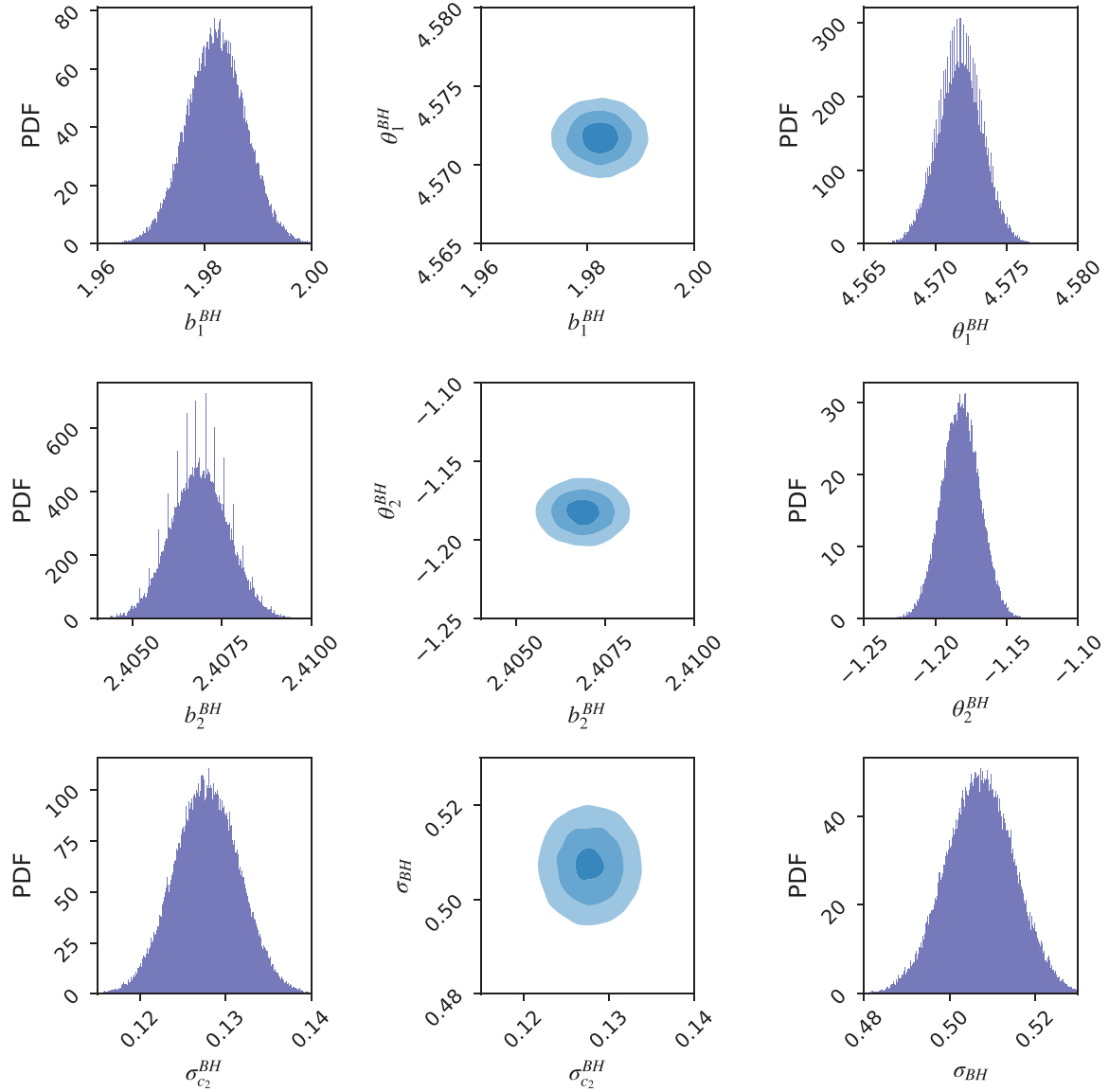


Figure 6.5.: MCMC-generated (§4.4.1) probability distributions for BH vs. c_2 (Equation (6.5)) model (top and middle rows) and slop/extrinsic scatter (bottom row) parameters. Parameter confidence ellipses (center) with 1-, 2- and 3σ regions show joint posterior probabilities with respect to the parameters plotted on either side. The pivot-point finding algorithm of §5.1 was used to remove respective correlations between model parameters of each linear "leg" of the model curve.

Input Data

Select Data Type:

Equal Weights (Default) Unequal Weights

Format: One data point per line: From left to right, list the independent (x) values followed by their error bars (uncertainties), then the dependent (y) values followed by their error bars, separated by white space.

Weights, if unequal, should be included on the same line, furthest to the right, again separated by white space.

1.7	0.2	2.64	0.4	1
2.1	0.2	3.59	0.4	1
2.56	0.2	4.58	0.4	1
5.65	0.2	4.53	0.4	1
4.83	0.2	4.47	0.4	1
6.56	0.2	7.28	0.4	1
7.72	0.2	8.57	0.4	1
8.28	0.2	8.74	0.4	1
7.74	0.2	9.57	0.4	1

Figure 6.6.: The data input step of the web-based TRK calculator with example data.

currently located at <https://skynet.unc.edu/rcr/calculator/trk>⁷. Easy to use, but full of customization options, the goal of this website was to create a simple, user-friendly introduction to the TRK statistic, including educational interactive visualization tools.

The first step of the web-based TRK calculator is to input the raw data, shown in Figure 6.6, where the user can give values for $\{x_n, \sigma_{x,n}, y_n, \sigma_{y,n}\}$ with optional weights $\{w_n\}$. Next, shown in Figure 6.7, the user can plot the data with error bars in order to estimate which model to fit to the data. The plot also offers various interactive tools, including the ability to pan and zoom⁸.

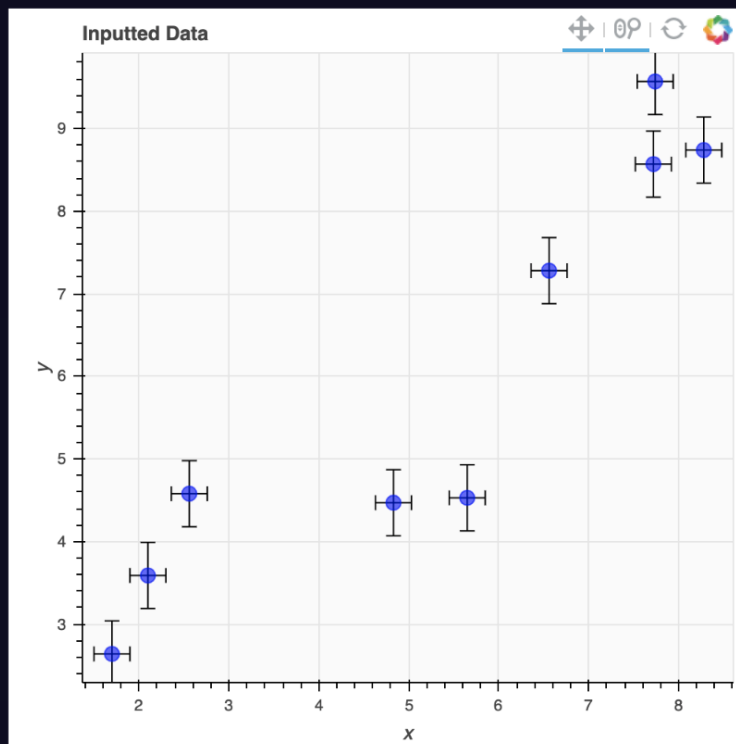
The following step is then for the user to choose the model to fit the data to, shown in Figure 6.8, and to provide an initial guess for the model and slope parameters for the fitting algorithm. The website comes with six built-in models: linear, quadratic, cubic, exponential, power law and logarithmic; the user also has the option to run the pivot-point finding/de-correlation algorithm of §5.1 on applicable models.

The last step of the calculator is for the user to choose which algorithms, if any, to run in addition to the regular likelihood-maximization downhill-simplex fitting of §4.2, shown in Fig. 6.9. The user can choose to either provide a fitting scale s (with a default of

⁷Note that this is part of the website that also includes the two Robust Chauvenet Outlier Rejection (RCR) calculators (see Maples et al. (2018)), that I am also the developer of (both the RCR source code and webpages). Long term, I plan to develop a single standalone statistical suite, available in multiple languages, that will bring together RCR, TRK, and other future statistical tools that I've developed, of which this website will be part of the ecosystem of.

⁸This plot, as well as all other plots on the TRK and RCR webpages, were made with Python's Bokeh library, from the Bokeh Development Team (2018).

Plot Data



Once the input data has been entered, plot it by mousing over the plot area. The plot can be panned over, and a zoom tool can be used by scrolling.

This can be used as a starting point to estimate the type of model to fit to the data.

Figure 6.7.: Example of plotting input data on the TRK calculator webpage.

Model Data

Model $y(x)$:

$$y(x) = a_0 + a_1(x-\bar{x})$$

Here, select a function to model $y(x)$. Depending on the chosen function, either x or $\log_{10} x$ is the "pivot point" depending on which model function is selected.

In the box to the right, enter your guess for the model function parameters a_0 , a_1 , a_2 , etc. For example, in the linear model case, a_0 corresponds to the y -intercept, and a_1 to the slope.

Finally, enter guesses for the slope (extrinsic scatter of the data) along the x axis and the y axis, as the last two lines (If unsure, guesses of about 1 are a safe bet) of this box.

For large datasets and/or for non-listed custom functions, it is recommended to use the C++ source code (documentation included) [here](#).

Input Initial Guess for Model Function Parameters (Including Slop):

0.9
1.1
0.5
0.3

Determine Pivot Point of Model (Optional):

No
(default)

Yes

If possible, by determining the pivot point of the model, the fitted model parameters can be made to be uncorrelated.

Figure 6.8.: Section of the TRK webpage where the user can choose which model to fit to their data.

Select Fitting Algorithm

Optimize Fitting Scale?

Yes (Default) No

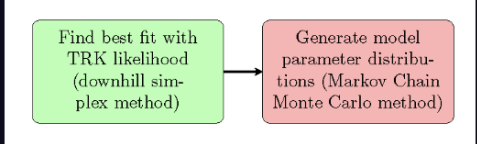
Select scale: (1.0 by default):

1.5

Generate Model Parameter Distributions/Uncertainties?

Yes No (Default)

Selected Fitting Algorithm:



```

graph LR
    A[Find best fit with TRK likelihood (downhill simplex method)] --> B[Generate model parameter distributions (Markov Chain Monte Carlo method)]
  
```

1. Find best fit at chosen scale by maximizing TRK Likelihood with downhill simplex method (§2.5.1, Trotter et al., in preparation)
2. Generate model parameter distributions using Adaptive Markov Chain Monte Carlo sampler (§2.5.2)

Figure 6.9.: Section of the TRK webpage where the user can determine which additional TRK algorithms to run alongside basic fit.

$s = 1.0$), or run the scale optimization algorithm. The user also has the option to use the MCMC algorithm of §4.4.1 to determine model parameter uncertainties; however, this is a computationally intensive process, so I am still experimenting with the possibility of expediting it. As shown, the total selected algorithm is displayed as a flow chart, with bullet-points below explaining the various steps⁹.

With these steps completed, the user can now perform a TRK Fit, in the section shown in Figure 6.10. After pressing the “Perform Fit” button and waiting for the algorithm to run, the final model and slop parameters are displayed on the left, as shown (with the model parameters ordered according to the functional form of the chosen model). If the scale optimization algorithm was run, the optimum and extreme scales, (s_0, a, b) , respectively, will also be displayed. The resulting fitted model curve is then plotted alongside the data, including 1-, 2- and 3σ confidence regions described by the fitted slop parameters (see Footnote 19 on page 39 to see how these regions are explicitly calculated).

⁹As shown, these bullet points also show section numbers; these correspond to the relevant sections of our upcoming paper, Trotter, Daniel E. Reichart, and Konz (2020), that will introduce the TRK statistic and some of its astrophysical applications in a peer-reviewed, journalistic context. The paper, currently in preparation, will be submitted to the Astrophysical Journal, Supplement Series.

6. Applications and Examples

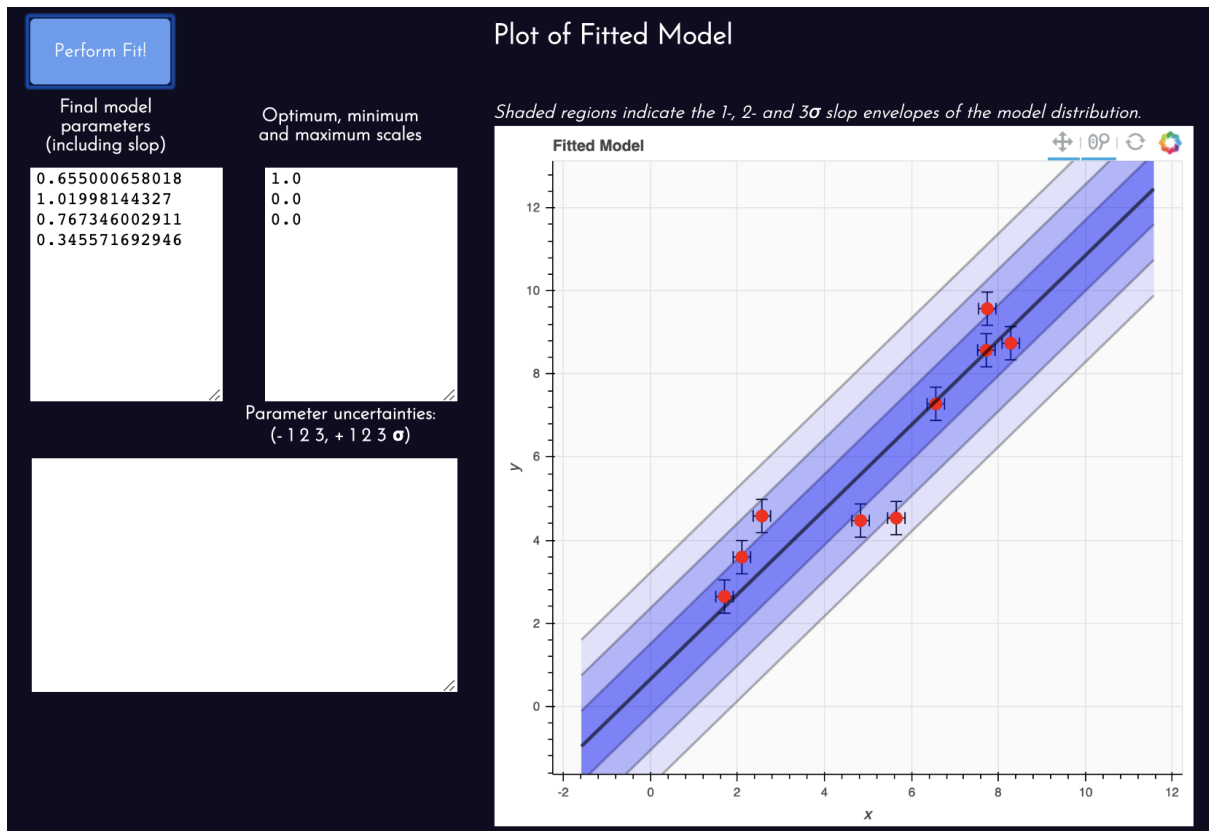


Figure 6.10.: Output section of the TRK calculator webpage, showing the results of an example linear fit (without model parameter uncertainty computation).

7. Future Endeavors

7.1. A Scale Optimization Algorithm for Asymmetric Uncertainties

Continuing from §5.2, the final consideration to be made involving the introduction of asymmetric error bars and/or slop is the *fit scale optimization* algorithm described in §4.3. Recall that in the symmetric case, this involves

1. determining the minimum scale $s = a$ where $\sigma_x \rightarrow 0$, and the maximum scale $s = b$ where $\sigma_y \rightarrow 0$, and
2. determining the optimum fitting scale $s_0 \in [a, b]$ by iteratively solving Equation (3.3).

However, in the asymmetric case there are *two* slop parameters along each dimension, so determining, or even qualifying, the minimum, maximum and optimum scales is nontrivial. I have attempted and/or posited a few potential methods of asymmetric scale optimization, and tested the scale-dependent best fit behavior of asymmetric fits, but the results have been inconclusive to date. As such, this issue remains as a future endeavor that will be explored in a later work.

7.2. Support for N -dimensional Models

For any statistic, it is desirable to be able to fit models to data with *multiple* independent (“ x ”) variables. I have considered the possibility of generalizing the TRK statistic to $N - 1$

7. Future Endeavors

independent variables (so N dimensions total) such that each datapoint could have N symmetric error bars, or up to $2N$ asymmetric error bars, and the model would have N parameters describing slop/extrinsic scatter. However, there are a number of practical issues that could make this leap difficult, or potentially intractable. First, the tangent-point finding algorithm of §2.2.2 and §4.1, that is required to simply evaluate the TRK likelihood of Equation (2.17), would require determining where some N -dimensional model curve is tangent to an N -dimensional error hyperellipsoid, which would not only conceivably greatly increase the computational power and possible issues with the method, but would require a totally new algorithm to account for the arbitrary number of dimensions. Even more fundamentally, the scale optimization routine of §4.3 would also have to be completely revamped, as not only is it entirely based off of 2-dimensional principles and algorithms, but the definition itself of fitting scale would have to change with addition of more dimensions, as adding N dimensions means there are now N possible ways to scale the dataset, described by N additional scale parameters. Overall, while an N -dimensional TRK statistic and algorithm would be extremely useful, and practically the ultimate general “worst-case” dataset model-fitting tool, developing it would likely prove to be very challenging, if even possible.

7.3. Python Implementation

I developed the TRK suite in C++ due to the language’s portability, low-level nature, and access to parallelization, to name a few reasons. However, Python is one of the most popular and widespread languages used within data science, the natural sciences, and other related statistically-based fields. Some of the most used model fitting algorithms are found within the many scientific Python libraries, such as SciPy (e.g. Virtanen et al. (2020)), and introducing the TRK suite to Python could make it accessible and useful to many more people, and make it possible to integrate and use the TRK statistic with many other Python codebases. As such, another long-term goal of this project is to develop/port the TRK suite into a standalone Python library. This could be done in many ways, but the main option I am currently considering is using SWIG (Simple Wrapper Interface Generator of Beazley et al. (1996)) to wrap the C++ TRK code into Python, which

7. Future Endeavors

I did at a basic level when developing the TRK calculator, as the website runs on a Python-based framework.

Appendix A.

Additional Algorithm Listings

Listed below are various algorithms that did not need to be given explicitly in the main text, but can be referenced here as needed.

Algorithm 10: Downhill simplex evolution functions used in Algorithm 4 to minimize $-2 \ln \mathcal{L}^{\text{TRK}}$.

```

1 Initialize simplex evolution parameters  $(\alpha, \beta, \gamma, \delta) = (1, 2, 0.5, 0.5)$ 
2 Function Expand
   Input : Simplex  $\Delta$  with  $\mathcal{M} + 1$  vertices  $v_i$ 
   Output: Expanded  $\Delta$ 
3   Compute expansion point of  $v_r$ ,  $v_e \equiv c + \gamma(v_r - c)$ .
4   if  $\chi_e^2 < \chi_r^2$  then
5       |  $v_{\mathcal{M}} \rightarrow v_e$ 
6   end
7   else
8       |  $v_{\mathcal{M}} \rightarrow v_r$ 
9   end
10  return  $\Delta$ 
11 Function Contract
   Input : Simplex  $\Delta$  with  $\mathcal{M} + 1$  vertices  $v_i$ 
   Output: Contracted  $\Delta$ 
12  Compute contraction point  $v_c$ , by using better of  $v_{\mathcal{M}}$ ,  $v_r$ .
13  if  $\chi_{\mathcal{M}-1}^2 \leq \chi_r^2 < \chi_{\mathcal{M}}^2$  then
14      | Contract Outside:
15      |  $v_c = c + \beta(v_r - c)$ 
16      | if  $\chi_c^2 \leq \chi_r^2$  then
17          |  $v_{\mathcal{M}} \rightarrow v_c$ 
18      | end
19      | else
20          | Shrink( $\Delta$ ) (See function below)
21      | end
22  end
23  else if  $\chi_r^2 \geq \chi_{\mathcal{M}}^2$  then
24      | Contract Inside:
25      |  $v_c = c + \beta(v_{\mathcal{M}} - c)$ 
26      | if  $\chi_c^2 < \chi_{\mathcal{M}}^2$  then
27          |  $v_{\mathcal{M}} \rightarrow v_c$ 
28      | end
29      | else
30          | Shrink( $\Delta$ )
31      | end
32  end
33  return  $\Delta$ 
34 Function Shrink
   Input : Simplex  $\Delta$  with  $\mathcal{M} + 1$  vertices  $v_i$ 
   Output: Shrunk  $\Delta$ 
35  for  $i = 0, \dots, \mathcal{M}$  do
36      |  $v_i \rightarrow v_0 + \delta(v_i - v_0)$ 
37  end
38  return  $\Delta$ 

```

Algorithm 11: Bracketing/Bisection-type method for determining maximum fitting scale b for some model and dataset.

```

1  Function FindMinimumScale
   Input : Model  $y_c$  and dataset  $\{x_n, y_n\}$  with error bars  $\{\sigma_{x,n}, \sigma_{y,n}\}$ 
   Output: Maximum fitting scale  $b$ .
2  begin
3      Determine brackets  $(l, r)$  for max scale  $b$ :
4      Initialize bisection brackets  $l = s = 0, r = s = 1$  and  $s_{\text{trial}} = s = 1$ 
5      Note that in the actual code, a better  $s_{\text{trial}}$  is found from the algorithm for
       finding  $a$ .  $\sigma_y(s_{\text{trial}}) \leftarrow \text{DownhillSimplex}(s = s_{\text{trial}})$ 
6      Initialize step modifier  $\alpha = 0.5 \times s_{\text{trial}}$ 
7      if  $\sigma_y(s_{\text{trial}}) > 0$  then
8           $l = s_{\text{trial}}$ 
9           $r_{\text{trial}} = s_{\text{trial}}$ 
10          $\sigma_y(r_{\text{trial}}) = \text{DownhillSimplex}(s = r_{\text{trial}})$ 
11         while  $\sigma_y(r_{\text{trial}}) > 0$  do
12              $r_{\text{trial}} = r_{\text{trial}} + \alpha$ 
13              $\sigma_y(r_{\text{trial}}) = \text{DownhillSimplex}(s = r_{\text{trial}})$ 
14              $l = r_{\text{trial}}$ 
15         end
16          $r = r_{\text{trial}}$ 
17     end
18     else if  $\sigma_y(s_{\text{trial}}) = 0$  then
19          $r = s_{\text{trial}}$ 
20          $l_{\text{trial}} = s_{\text{trial}}$ 
21          $\sigma_y(l_{\text{trial}}) = \text{DownhillSimplex}(s = l_{\text{trial}})$ 
22         while  $\sigma_x(l_{\text{trial}}) = 0$  do
23              $l_{\text{trial}} = l_{\text{trial}} - \alpha$ 
24              $\sigma_y(l_{\text{trial}}) = \text{DownhillSimplex}(s = l_{\text{trial}})$ 
25              $\alpha = 0.5 \times \alpha$ 
26              $r = l_{\text{trial}}$ 
27         end
28          $l = l_{\text{trial}}$ 
29     end
30     Use bisection to determine  $b$  now that we have brackets  $(l, r)$ :
31      $b_{\text{trial}} = (l + r)/2$ 
32      $\sigma_y(b_{\text{trial}}) = \text{DownhillSimplex}(s = b_{\text{trial}})$ 
33     while  $|l - r| \geq \text{tolerance1}$  AND  $\sigma_y(a_{\text{trial}}) \geq \text{tolerance2}$  do
34          $b_{\text{trial}} = (l + r)/2$ 
35          $\sigma_y(b_{\text{trial}}) = \text{DownhillSimplex}(s = b_{\text{trial}})$ 
36         if  $\sigma_y(b_{\text{trial}}) > 0$  then
37              $l = b_{\text{trial}}$ 
38         end
39         else if  $\sigma_y(b_{\text{trial}}) = 0$  then
40              $r = b_{\text{trial}}$ 
41         end
42     end
43     return  $b = b_{\text{trial}}$ 
44 end

```

Algorithm 12: Bisection-type method for determining optimum fitting scale s_0 for some model and dataset with minimum and maximum fitting scales a and b .

```

1 Function FindOptimumScale
   Input : Model  $y_c$  and dataset  $\{x_n, y_n\}$  with error bars  $\{\sigma_{x,n}, \sigma_{y,n}\}$ , with
           minimum and maximum fitting scale  $a$  and  $b$ .
   Output: Optimum fitting scale  $s_0$ .
2 begin
3   Initialize  $s_0^{(1)} = (a + b)/2$ 
4   while  $|s_0^{(2)} - s_0^{(1)}| \geq \text{tolerance1}$  do
5      $s_0^{(1)} = s_0^{(2)}$ 
6     Use bisection to determine  $s_0^{(2)}$  given  $s_0^{(1)}$ :
7     Initialize brackets  $l = a, r = b$ 
8      $s_{0,\text{trial}}^{(2)} = (l + r)/2$ 
9      $R_{\text{trial}} = \text{Equation (4.1)} \leftarrow \text{DownhillSimplex}(s = s_{0,\text{trial}}^{(2)})$ 
10    while  $|l - r| \geq \text{tolerance2}$  AND  $|R| \geq 0$  do
11       $s_{0,\text{trial}}^{(2)} = (l + r)/2$ 
12       $R_{\text{trial}} = \text{Equation (4.1)} \leftarrow \text{DownhillSimplex}(s = s_{0,\text{trial}}^{(2)})$ 
13       $R_l = \text{Equation (4.1)} \leftarrow \text{DownhillSimplex}(s = l)$ 
14      if  $R_{\text{trial}} \times R_l > 0$  then
15         $l = s_{0,\text{trial}}^{(2)}$ 
16      end
17      else if  $R_{\text{trial}} \times R_l < 0$  then
18         $r = s_{0,\text{trial}}^{(2)}$ 
19      end
20    end
21     $s_0^{(2)} = s_{0,\text{trial}}^{(2)}$ 
22  end
23  return Optimum scale  $s_0 = s_0^{(2)}$ 
24 end

```

Bibliography

- (<https://stats.stackexchange.com/users/63677/forgottenscience>), Forgottenscience (2020). *How do I account for numerical overflow with Adaptive MCMC?* Cross Validated. URL:<https://stats.stackexchange.com/q/453533> (version: 2020-03-10). eprint: <https://stats.stackexchange.com/q/453533>. URL: <https://stats.stackexchange.com/q/453533> (cit. on p. 34).
- Beazley, David M et al. (1996). “SWIG: An Easy to Use Tool for Integrating Scripting Languages with C and C++.” In: *Tcl/Tk Workshop*. Vol. 43 (cit. on p. 74).
- Bickel, David R and Rudolf Fruehwirth (2005). “On a fast, robust estimator of the mode: Comparisons to other robust estimators with applications.” In: *arXiv preprint math/0505419* (cit. on p. 45).
- Bokeh Development Team (2018). *Bokeh: Python library for interactive visualization*. URL: <https://bokeh.pydata.org/en/latest/> (cit. on p. 68).
- Brown, Philip J, Wayne A Fuller, et al. (1990). *Statistical analysis of measurement error models and applications: Proceedings of the AMS-IMS-SIAM joint summer research conference held June 10-16, 1989, with support from the National Science Foundation and the US Army Research Office*. Vol. 112. American Mathematical Soc. (cit. on p. 56).
- Cardelli, Jason A, Geoffrey C Clayton, and John S Mathis (1989). In: *The Astrophysical Journal* 345, pp. 245–256 (cit. on p. 59).
- Clayton, Geoffrey C et al. (2015). “New ultraviolet extinction curves for interstellar dust in M31.” In: *The Astrophysical Journal* 815.1, p. 14 (cit. on pp. 59, 62, 66).
- D’Agostini, G (2005). “Fits, and especially linear fits, with errors on both axes, extra variance of the data points and other complications.” In: *arXiv preprint physics/0511182* (cit. on p. 57).

Bibliography

- Fitzpatrick, Edward L and Derck Massa (1988). In: *The Astrophysical Journal* 328, pp. 734–746 (cit. on p. 59).
- Fitzpatrick, EL and D Massa (2007). “An analysis of the shapes of interstellar extinction curves. V. The IR-through-UV curve morphology.” In: *The Astrophysical Journal* 663.1, p. 320 (cit. on pp. 59, 62, 66).
- Gordon, Karl D et al. (2003). “A quantitative comparison of the Small Magellanic Cloud, Large Magellanic Cloud, and Milky Way ultraviolet to near-infrared extinction curves.” In: *The Astrophysical Journal* 594.1, p. 279 (cit. on pp. 59, 62, 66).
- Haario, Heikki, Eero Saksman, Johanna Tamminen, et al. (2001). “An adaptive Metropolis algorithm.” In: *Bernoulli* 7.2, pp. 223–242 (cit. on p. 36).
- Hastings, W Keith (1970). “Monte Carlo sampling methods using Markov chains and their applications.” In: (cit. on p. 33).
- Isobe, Takashi et al. (Nov. 1990). “Linear Regression in Astronomy. I.” In: 364, p. 104 (cit. on p. 57).
- Maples, MP et al. (2018). “Robust Chauvenet Outlier Rejection.” In: *The Astrophysical Journal Supplement Series* 238.1, p. 2 (cit. on pp. 27, 68).
- Morrison, Faith A (2014). “Obtaining uncertainty measures on slope and intercept of a least squares fit with Excel’s LINEST.” In: *Houghton, MI: Department of Chemical Engineering, Michigan Technological University. Retrieved August 6, p. 2015* (cit. on p. 44).
- Nelder, J. A. and R. Mead (1965). In: *Computer Journal* 7, pp. 308–313 (cit. on p. 27).
- Pearson, Karl (1896). “VII. Mathematical contributions to the theory of evolution.—III. Regression, heredity, and panmixia.” In: *Philosophical Transactions of the Royal Society of London. Series A, containing papers of a mathematical or physical character* 187, pp. 253–318 (cit. on p. 16).
- Reichart, Daniel E (2001). “Dust extinction curves and Ly α forest flux deficits for use in modeling gamma-ray burst afterglows and all other extragalactic point sources.” In: *The Astrophysical Journal* 553.1, p. 235 (cit. on pp. 57, 58).
- Tiruneh, Ababu Teklemariam, William N Ndlela, and Stanley J Nkambule (2013). “A two-point Newton method suitable for nonconvergent cases and with super-quadratic convergence.” In: *Advances in Numerical Analysis* 2013 (cit. on p. 22).

Bibliography

- Trotter, Adam S. (2011). “The Gamma-Ray Burst Afterglow Modeling Project: Foundational Statistics and Absorption & Extinction Models.” PhD thesis. The University of North Carolina at Chapel Hill (cit. on pp. [iv](#), [1](#), [2](#), [6–10](#), [12–19](#), [21](#), [29](#), [36](#), [39](#), [41](#), [42](#), [46](#), [49–52](#), [57–61](#), [64](#)).
- Trotter, Adam S., Daniel E. Reichart, and Nicholas C. Konz (2020). “Fitting Models to Data With Both Intrinsic and Extrinsic Uncertainties in Two Dimensions: Extinction and Absorption Models and the LV Relation.” In: Publication in preparation (cit. on pp. [iv](#), [71](#)).
- Valencic, Lynne A, Geoffrey C Clayton, and Karl D Gordon (2004). “Ultraviolet extinction properties in the Milky Way.” In: *The Astrophysical Journal* 616.2, p. 912 (cit. on pp. [59](#), [62](#), [66](#)).
- Virtanen, Pauli et al. (2020). “SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python.” In: *Nature Methods* 17, pp. 261–272. DOI: <https://doi.org/10.1038/s41592-019-0686-2> (cit. on p. [74](#)).