# 1 Parameter Estimation

We find the parameters $\mathbf{x}$ that solve the inverse problem $G(\mathbf{x}) = \mathbf{d}$, using a direct search method (*Nelder-Mead*).

$$\min\ \chi^2 = \min\ \sum_{i=1}^{n} \frac{(y_d(t)_i - y_G(t,\mathbf{x})_i)^2}{y_G(t,\mathbf{x})_i}$$

$$\mathbf{x} = \begin{cases} q \\ V_d \\ \sigma \end{cases} \quad \text{subject to constraints} \quad g = \begin{cases} V_d & \pm\ u_{exp} \\ \sigma & \pm\ u_{exp} \\ y_0 & \pm\ u_{exp} \\ t_0 & \pm\ u_{exp} \end{cases}$$

where $y_G(t,\mathbf{x})$ is a numerical solution of the equation of motion

$$my'' = \frac{1}{2}\rho C_D A_d y'^2 + qE(y) + \frac{1}{4}\frac{Kq^2}{y^2} + \frac{1}{2}E(y)^2\nabla\epsilon$$

In particular, it is impractical to directly measure the droplet net charge $q$, during a bounce experiment. Our work flow to identify this parameters is as follows:

1. Experimentally vary $V_d$, $\sigma$ and capture droplet trajectories using a high-speed camera.

2. Digitize droplet trajectories by using automatic tracking of ellipse-fitted centroids on the thresholded video.

3. Slice droplet trajectories by their bounce minima, and apply a smoothing filter.

4. Extract the droplet charge (and other experimental parameters) by maximizing the log-likelihood of the data given the dynamical model and parameters, by varying the parameter vector using a direct search optimization.

## 1.1 Inverse Problems

We have a simple model for 1D projectile motion of charged droplets within Coulomb potential wells in low-g. Using various scaling arguments we have gleaned from this model a series of non-dimensional numbers characteristic of droplet bounce apoapses and times of flight, but these dimensionless groups depend on a set of physical parameters. Unfortunately not all of these parameters are physically practical to accurately measure by experiment. Droplet net charge $q$, in particular, could in principle be directly measured by collecting the charged drops in a faraday cup under low-g and measuring the change in capacitence of the cup using a very high input-resistence electrometer, but this is a problematic experiment to set up in a drop tower from a practical standpoint. The other state variables we can directly measure by experiement with varying levels of accuracy. To measure the charge, $q$, we instead turn to parameter estimation techniques.

Suppose we have a model $G(\mathbf{x})$, with a vector of parameters $\mathbf{x}$, and set of (noiseless) observations $\mathbf{d}$, the we naturally expect there to exist a relationship

$$G(\mathbf{x}) = \mathbf{d},$$

where the operator $G$ might be an ODE. Suppose the model $G(\mathbf{x})$ is the ODE

$$\frac{dy}{dt} = f(t, \mathbf{y}; \mathbf{x}),\ \mathbf{y} \in \mathbb{R}^n,$$

and a collection of $n$ measurements of experimental data

$$\mathbf{d} = (t_1, \mathbf{y_1}), (t_2, \mathbf{y_2}), ..., (t_k, \mathbf{y_k}).$$

The process of fitting a function, defined by a collection of parameters, to a data set is called the discrete inverse, or parameter estimation problem (as opposed to the *forward problem* to find $\mathbf{d}$ given $\mathbf{x}$ and $G(\mathbf{x})$). This is a familiar procedure when the determination of model parameters is done using linear or polynomial

regression. However there are approaches even to fitting an arbitrary function to a noisy and sparse dataset. In this work we use the conventional Maximum Likelihood Estimate (MLE) method to identify the model parameters.

Using MLE we don't ask the question: "what is the probability that my set of model parameters is correct?"(because the probability is very nearly zero!), but rather "given my set of model parameters, what is the probability that this data set occurred (what is the likelihood of the parameters given the data)?". Bayes' Theorem holds that

$$\text{prob}(X|D,I) = \frac{\text{prob}(D|X,I) \times \text{prob}(X|I)}{\text{prob}(D|I)}$$

where $D$ are our observations (dataset), $X$ is our vector of parameters, and $I$ is general background information about the problem including our mathematical model (for instance the ODE above), and

| | |
|---|---|
| $\text{prob}(X\|D,I)$ | posterior probability density function, |
| $\text{prob}(D\|X,I)$ | likelihood function, |
| $\text{prob}(X\|I)$ | prior probability density function, |
| $\text{prob}(D\|I)$ | evidence. |

The posterior probability density function (PDF) $\text{prob}(X|D,I)$, is ultimately what we want to estimate, the prior PDF $\text{prob}(X|I)$, reflects our knowledge of the system, and the evidence $\text{prob}(D|I)$, is the likelihood of the data based on our knowledge. We also note that since it only makes sense to compare the conditional PDF's for the same data, we can ignore the denominator (that is, the evidence). We further note that the prior $\text{prob}(X|I)$, is fixed before our observations and so can be treated as invariant to our problem. We can therefore infer that $\text{prob}(X|D,I) \propto \text{prob}(D|X,I)$. The MLE for the the model parameters $\mathbf{x_0}$, then is given by the maximum of the posterior PDF, which is equivalent to the solution of the ODE given the the parameters $\mathbf{x}$, that produces the highest probability of the observed data. Since the likelihood $\mathcal{L}(\mathbf{x}) = \prod_i^n \mathcal{P}_i$,

and the probability $\mathcal{P}$, of any single observation is less than one, then the total likelihood which is the product of a large number of probabilities tends to be vanishingly small. The more well behaved log-likelihood is given by

$$\mathcal{M} = \ln(\mathcal{L}) = \ln(\text{prob}(D|X, I)) = \text{const} - \frac{\chi^2}{2}$$

where

$$\chi^2 = \sum_{i=1}^{n} \frac{(y_{di} - y_{Gi})^2}{\sigma_{d_i}^2}$$

is the $\chi^2$ goodness-of-fit, $y_d = y_d(t)$ is a observation of droplet position at a point in time, and $y_G = y_G(t, \mathbf{x})$ is the droplet position predicted by the solution to the equation of motion at time $t$, and $\sigma$ is the standard error of the position measurement. If the number of data points $n$, is small was can use the Poisson error $\sigma_d^2 = y_G$. The optimal parameter set is the one with the highest probability of observing the data (the maximum of the posterior PDF) and can be determined by maximizing the log-likelihood $\mathcal{M}$ (or minimizing $\chi^2$) of the data $\mathbf{d}$ with respect to the parameter set $\mathbf{x}$. Thus parameter estimation is a variety of optimization problem.

## 1.2 Optimization

Most generally a constrained optimization problem is stated as

$$\begin{array}{lll} \text{minimize:} & f(\mathbf{x}) & \text{objective function} \\ \text{subject to:} & & \\ & g_j(\mathbf{x}) \leq 0 & \text{inequality constraints} \\ & h_k(\mathbf{x}) = 0 & \text{equality constraints} \end{array}$$

$$\text{where } \mathbf{x} = \begin{Bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{Bmatrix} \quad \text{design variables}$$

Mathematical optimization is the problem of finding minima of a function $f$. In this context the function is called the cost, or objective function. The field of mathematical optimization is as old as calculus itself, and the number of particular optimization techniques is correspondingly myriad; particular techniques lend themselves well to particular types of optimization problems. The minima of the objective function $f$ is sought on a domain $A$ specified by the constraints of the problem; this domain is usually called the feasible region. Minima of objective function $f : A \to \mathbb{R}^m$ are called feasible solutions. If the function $f$ is convex the feasible solution is the global minimum, otherwise additional local minima exist. The scale of the optimization problem is set ultimately by dimensionality of the objective function. Functions may not always be smooth in the sense of having continuous derivatives, and this is problematic in that optimization methods fundamentally rely on gradients of the objective function. Problems with anisotropic objective functions where there is strong covariance between the parameters, and the gradient vector generally to differ significantly from the Newton direction ($-\mathbf{H}^{-1}f'(\mathbf{x})^T$, where $\mathbf{H}$ is the Hessian matrix) are considered ill-conditioned. Ill-conditioned problems gradient based deterministic search tend to converge slowly as they take a zigzagging path determined by the local value of the gradient rather than following the Newton-direction vector which towards the minimum. Numerical optimization may deal with black box functions (where we do not have an explicit mathematical expression of the function we are optimizing). Black box problems are challenging because we do not have access to analytic gradients of the objective function, and approximating them by finite-differences is slow and noisy. In general, noisy, black box, non-linear, non-quadratic, non-convex, constrained, ill-conditioned, high-dimensional objective functions are problematic to optimize. Unfortunately, problems of this type are the essence of the parameter estimation, which often leads to its characterization as an 'art' rather than a precise science (though we submit that it is a dark art).

The equation of motion behaves stiffly due to the large disparity in Coulom-

bic, image charge, and dielectrophoretic length scales. We integrate it numerically using the `odeint` *Scipy* module. This is a shake-and-bake Python wrapper for the venerable 1982 *netlib ODEPACK* library double-precision `lsoda` (Livermore Solver for Ordinary Differential equations with Automatic method switching for stiff and nonstiff problems) integrator [ref]. The function switches between Adams (nonstiff) and Backwards Differentiation Formulas (BDF, stiff) according to the dynamic value of a set of stiffness eigenvalues.

Our specific optimization problem is non-convex, mixed discrete-continuous black-box (noisy), and highly ill-conditioned which is essentially the worst the worst case scenario for an optimization problem. The ill-conditioning arises due to the strong covariance between several of the model parameters (particularly $q = q(V_d, E_0)$). The non-convexity of the problem implies that there are many local minima of the objective function. While in principle a gradient-based optimizer (for instance using the quasi-Newton method of Broyden, Fletcher, Goldfarb, and Shanno (BFGS) [Nocedal, J, and S J Wright. 2006]) could be used by using finite-differences to obtain approximate gradients of the $\chi^2$ objective function, in practice doing so is extremely problematic because the noise-to-signal ratio of the objective function scales like $\mathcal{O}(f)$ for $\frac{df}{dt}$ and $\mathcal{O}(f^2)$ for $\frac{d^2f}{ft^2}$ which will tend to cause convergence to a local minima with is an artifact of the likelihood response surface [Wood, 1982: Refs 5, 14, 49, 95.]. As a further practical matter, given the relatively expensive function-calls (which requires solving a stiff, non-linear ODE) gradient-free approaches tend to offer better performance regardless[Singer and Mead, 2009].

We use a gradient-free, direct-search approach: Nelder-Mead [Nelder, 1965] implemented in `scipy.optimize` [Jones et al. 2001 –]. Nelder-Mead is robust to noise (relatively speaking), and is relatively thrifty with our extremely expensive function-calls. *Nelder-Mead*, sometimes called simplex-search or downhill-simplex, is a heuristic search method, with no guarantee of optimal solutions, but is well-established and widely used despite that. *Nelder-Mead* is based on the concept of a $N$-simplex, which generalizes a triangle into higher dimensions

as a polytrope of $N+1$ vertices in $N$ dimensions. It uses only-function calls and expands or contracts the simplex according to the function values at its vertices in a way visually reminiscent (in $\mathbb{R}^2$) of the oscillations of the jumping droplets themselves (in fact *Nelder-Mead* is sometimes also called the "amoeba method"). Very little is known about the convergence properties of the *Nelder-Mead* algorithm in its classical form for non smooth objective functions ([Price and Coope, 2003]), except that in general it doesn't satisfy the properties required for convergence by other direct search algorithms: that the simplex remains uniformly non-degenerate, and that some form of "sufficient" descent condition for function values at the vertices is required at each iteration. Scaling can help solve convergence problems and improve numerical stability. We precondition the optimization problem by minimizing $\ln(\chi^2)$, and using a naive scaling (scaling variables such that their magnitudes $\sim 1$) of our constraints by their initial guesses. Here is goal is to make the problem equally sensitive to steps in any direction. *Nelder-Mead* is not a global optimizer, though there are variants which use sequential local searches with probabilistic restarts to achieve globality. However global optimization usually comes at a tremendous computational cost. However, *Nelder-Mead* behaves less locally than many gradient-based approaches. The convergence history of the parameter MLE using *Nelder-Mead* for a single drop jump experiment is shown in Figure **??**.

## 1.3 Smoothing

All optimization methods, including *Nelder-Mead*, whether explicit or implicitly follow gradients towards an optimum. In a parameter estimation problem, if we approximate these gradients by finite differences, then the noise manifests itself as amplification of the roughness in the hyper-response surface. Gradient based optimizers do poorly in these situations because they tend to converge to local minima. While so called gradient free algorithms offer an improvement in this regard, speed of convergence and the quality of the MLE is improved by smoothing the objective function. This is equivalent to smoothing the underly-

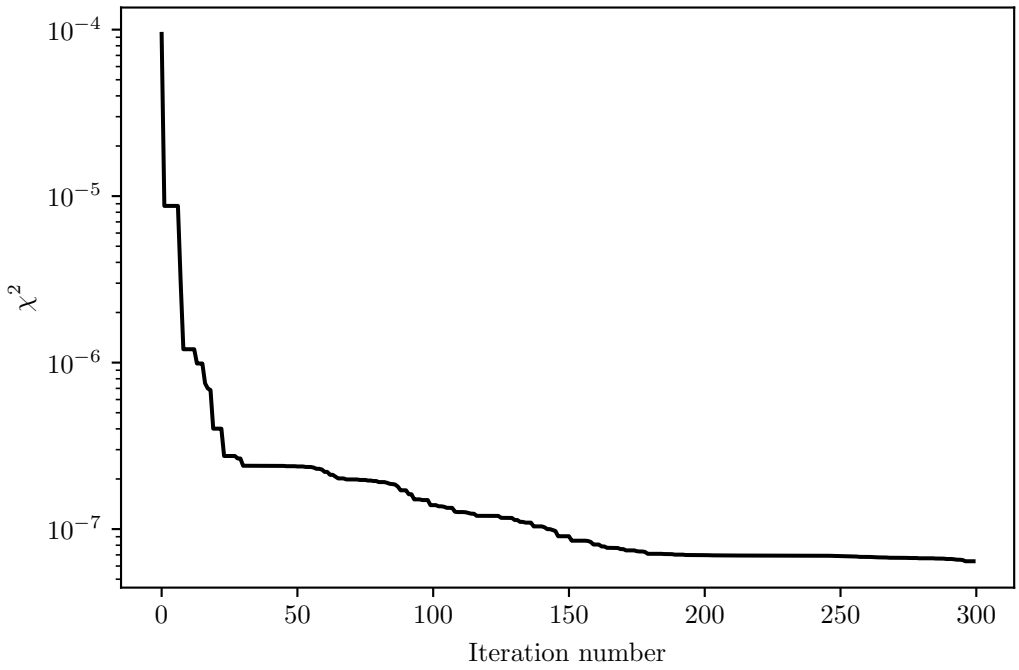


Figure 1: As is typical with *Nelder-Mead* much of the improvment in $\chi^2$ is realized in the first few iterations. Overall the rate of convergence is sub-linear, which is to be expected for non-liner constrained problems using an hueristic algorithem.

ing dataset.

Our choice of smoothing approach depends principally on the nature of the errors in the dataset. The sources of error include misalignment of the camera, error in the fiduciary length scale, perspective due to objects (subject or reference scale) being out of the photographic plane, and various errors arising in the digitization process (including the difference between the thresholded ellipse fitted centroid and the true centroid of the non-ellipse drop centroid). Some of these errors are systematic in origin and introduce consistent biases into the data (e.g. coherent spectral sources, rather than truly stochastic noise). Data smoothing does little to help systematic errors in that they are usually of lower frequency than the signal. Random errors, by contrast, are assumed to have a Gaussian distribution (by the central limit theorem), and are independent of the signal (which inherently results from a deterministic process).

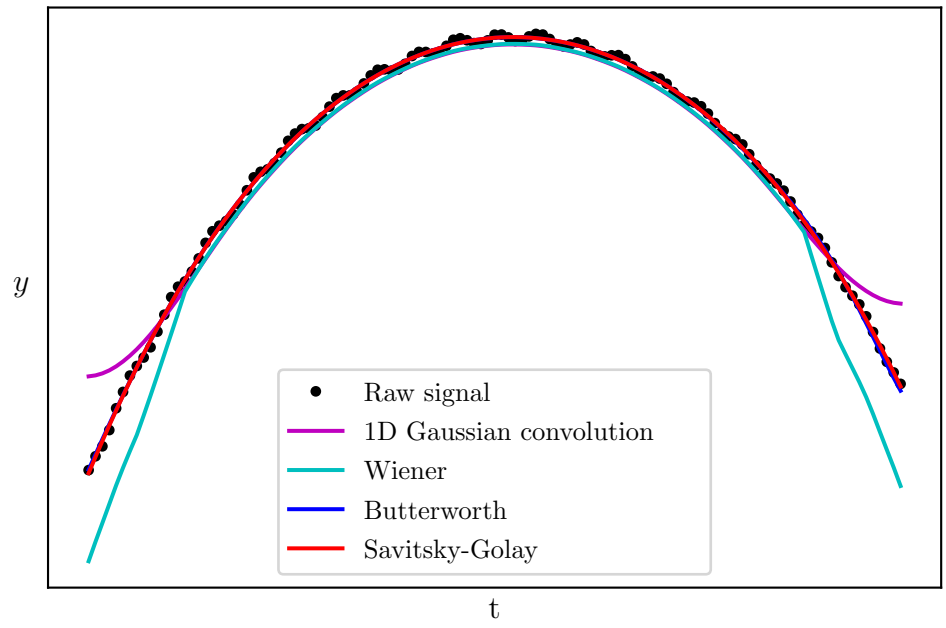


Figure 2: The underlying signal is 'noisy', due partially to deterministic errors in determining the centroid position. These deterministic errors are largely due to droplet oscillations, especially the rapidly damped higher harmonics which do not have azimuthal symmetry. There is also Gaussian error in the ellipse fitting due to thresholding and noise in the video itself. We see that 1D Gaussian convolution and Wiener filters suffer from significant end effects. At this scale Butterworth and Savistsky-Golay filters are nearly indistinguishable.

We experimented with a variety of filters implemented in the `scipy.signal` *SciPy* module on a representative set of trajectory data; these methods include 1D Gaussian convolution, Wiener, Butterworth, and Savitsky-Golay filters. Qualitatively comparing these smoothing methods (by hand tuning filter orders and window sizes) we find that we loose too many data points in the smoothing process, large amplitudes are overly smoothed by repeated filtering passes, or there are significant end effects for most of these methods. A comparison of these smoothing approaches on a representative trajectory data set are shown in Figure **??**.

The Savitsky-Golay, and Butterworth filters both produce fairly smooth derivatives as can be seen in Figure **??**; but the small window-size needed for Butterworth filter tends to also produce a noticeable end effect. The Savitsky-

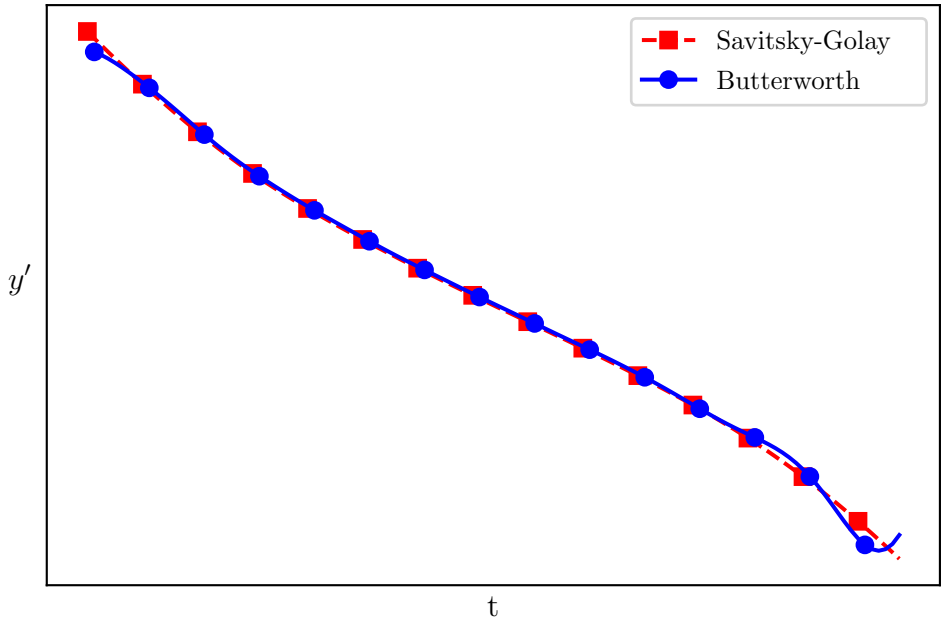


Figure 3: Comparing the first derivatives of the Butterworth and Savitsky-Golay filters we see that the Butterworth filter also suffers from a slight end effect. This implies that the optimization will find a different optima of the likelyhood depending on which type of filter is used.

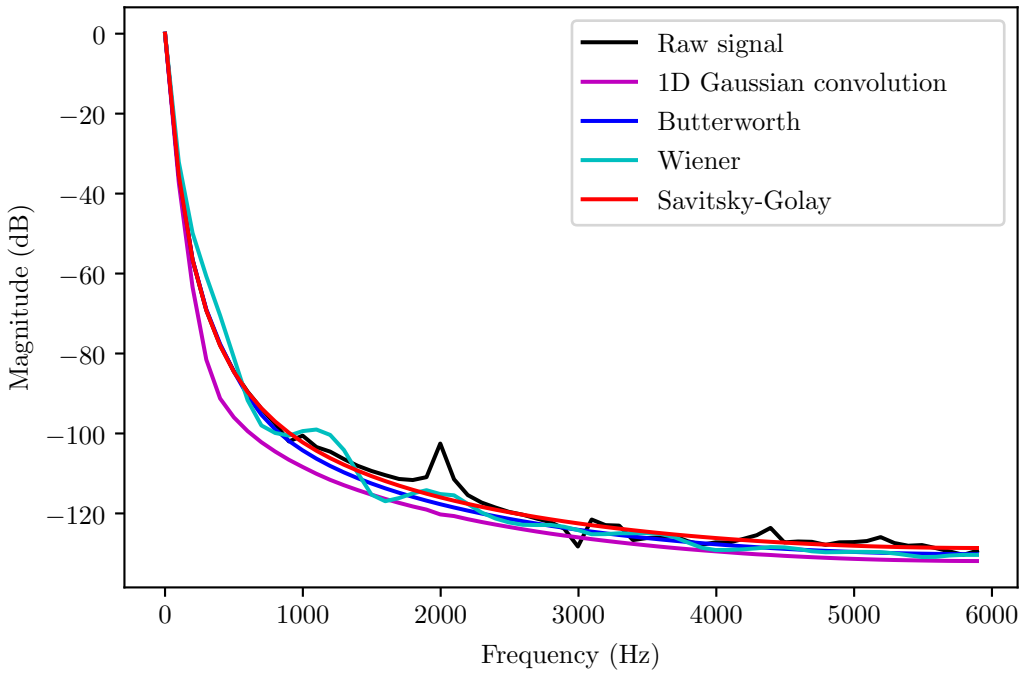


Figure 4: The power spectra has a peak at 1 Hz, which is the droplet trajectory parabola itself, and smaller peaks in the kHz range corresponding to various noise frequencies. The Savitsky-Golay and Butturworth filters seem to have the least distortion of the power spectra of the true signal. Both also do a good job of attenuating the noise at the 2 kHz peak.

Golay filter essentially uses a moving-window based on local least-squares polynomial approximations. It was shown that fitting a polynomial to a set of input samples and then evaluating the resulting polynomial at a single point within the approximation interval is equivalent to discrete convolution with a fixed impulse response [Savitsky, 1964]. A beneficial property of this kind of low-pass filter is their tendency to maintain waveform amplitudes, and so they are attractive in applications having noisy signals with sharply pointed waveforms such as ultrasound or synthetic aperture radar [Schafer, 2011]. Because Savitsky-Golay is a Finite Impulse Response (FIR) filter it requires data points to be equally spaced; to accommodated this we interpolate points between the small gaps which sometime occur in the tracking results from image analysis. We use a moving window size slightly smaller than the length of the current bounce in a drop jump data set. The windows are piecewise defined by partitioning the data set into a series of individual bounces (the dataset is sliced at minima identified after an initial rough smoothing pass, using the `scipy.signal.argrelextrema()` function). The Savitsky-Golay polynomial order is 4. To understand how these filters differ it is useful to look at their frequency response. In Fourier space, convolution becomes a multiplication, and we can understand what a filter does by looking at which frequencies it lets pass through. We can do this using a Discrete Fourier Transform (though it is worth noting that our signal is not truly periodic). The power spectra for these same data are shown in Figure **??**.

## 1.4 Identifiability

That we are capable of fitting any arbitrary model to a dataset given sufficient degrees of freedom in our parameters is admittedly a disconcerting issue, begging the question "given the structure of the model is it possible to uniquely estimate the unknown parameters?" This question is called the problem of identifiability. However, some of the inverse model parameters are constrained by our experimental observations of them and their associated measurement uncertainties. This, we hope, makes the specter of an over fitted model less frightening, but

does convert our unconstrained optimization problem to an constrained one which raises special difficulties of its own, which we discuss below.

We're interested in the variance and co-variance as a means to determine the quality if the parameter estimate. The $(i, j)$-th element of the matrix $\sigma(\mathbf{x}, \mathbf{y})$ is equal to the covariance $\mathrm{cov}(X_i, Y_j)$ between the $i$-th scalar component of $\mathbf{x}$ and the $j$-th scalar component of $\mathbf{Y}$. Here the concept of error bars in linear correlation associated with a covariance matrix are not suitable. We might try to generalize the idea of confidence intervals to a multidimensional space, but usually it will be hard to describe the surface of the (smallest) hyper-volume containing 90% of the probability in just a few numbers. The situation is worse if the probability density function has several maxima. However we notice that

$$\left[\sigma^2\right]_{ij} = -\left[(\nabla\nabla\mathcal{L})^{-1}\right]_{ij} = 2\left[\nabla\nabla\left(\chi^2\right)\right]_{ij}^{-1} = -\left[H^{-1}\right]_{ij}$$

where $H$ refers to the Hessian matrix and $\left[\sigma^2\right]_{ij}$ is the covariance matrix $C$. The issue of identifiability is especially fraught for non-linear, black box type problems were it is difficult to explicitly evaluate the Hessian. The likelihood function (and thus the posterior probability density function) are defined completely by the optimal solution $\mathbf{x}$ and the second derivative of $\mathcal{L}$ at the maximum, which corresponds to the covariance matrix $C$. The standard errors (marginal variances) are the square roots of the diagonal of the covariance matrix. Our relative errors thus produced are extremely small ($> 1\%$). The Hessian matrix must be negative definite for $\mathcal{L}$ to have a maximum at $\mathbf{x_0}$. We can use the condition number

$$\mathrm{cond}(A) = \frac{\max[\mathrm{eig}(A)]}{\min[\mathrm{eig}(A)]}$$

as a means to evaluate to stability of our problem. We find typical condition numbers $\sim \mathcal{O}(10^{27})$ which indicate the problem is strongly ill-conditioned near the mimima $\mathbf{x_0}$.

The *Nelder-Mead* direct search method cannot be used with explicitly constrained problems. However, there are various implicit approaches to (approx-

imately) solving general constrained problems using unconstrained algorithms. Generally, this is achieved by domain transformations or the use of penalty functions. By the addition of a penalty function which depends in some way on the values of the constraints to the objective function, we minimize a pseudo-objective function where the in feasibility of the constraints is minimized simultaneously to the objective function.

There are various penalty function schemes. We use an Exterior Penalty Function as a simple way of converting converting the constrained problem into an unconstrained one. These are are especially useful in cases where the constraints are not "hard" in the sense that they need to be satisfied precisely. General penalty functions, which are sequential unconstrained minimization techniques, reformulate the general constrained problem as the pseudo-objective function given by

$$\phi(\mathbf{x}, r_p) = F(\mathbf{x}) + r_p P(\mathbf{x})$$

where $P(\mathbf{x})$ the penalty function, is given by

$$P(\mathbf{x}) = \sum_{j=1}^{m} \left\{\max\left[0, g_j(\mathbf{x})\right]\right\}^2 + \sum_{k=1}^{l} \left[h_k(\mathbf{x})\right]^2. \tag{1}$$

We see from Equation **??** that there is no penalty if the constraints $g_j(\mathbf{x})$, and $h_k(\mathbf{x})$ are satisfied.

The Exterior Penalty Function specifically (and all Penalty Function approaches in general) do have several drawbacks. Namely these include the possibility of the objective function being undefined outside of the set of feasible solutions. Additionally, by naively "encouraging" feasibility of the solution using large values of the penalty parameter, $r_p$, we will tend to ill-condition the unconstrained formulation of the problem (though in our implementation the preconditioning tends to make the pseudo-objective function less and less sensitive to the constraints as the likelihood is approaches a maximum). We use the the measured values of $u_0$, $V_d$, and $E_0$, and the informed guess $q \approx kV_d^{2/3}E_0$

where k is a constant $k \approx 1^{-11}$ as an initial guess for the parameter vector $\mathbf{x}$. We stop the optimization after 300 iterations rather than waiting for convergence.

Despite their wide use in parameter estimation, there is very little known about convergence properties of *Nelder-Mead*. The iterative procedure may thus converge very slowly near the optimum, unless the Hessian is "well-conditioned", i.e., the condition number is near one. However, if parameter estimates are highly correlated among themselves, the Hessian matrix is near-singular and its inversion is either impossible or involves significant numerical error. Posteriori verification of the results is crucial to bound the identifiability of the parameter estimation problem, yet it has not be done yet. [Ref] suggests verification by generation of Monte Carlo data sets.

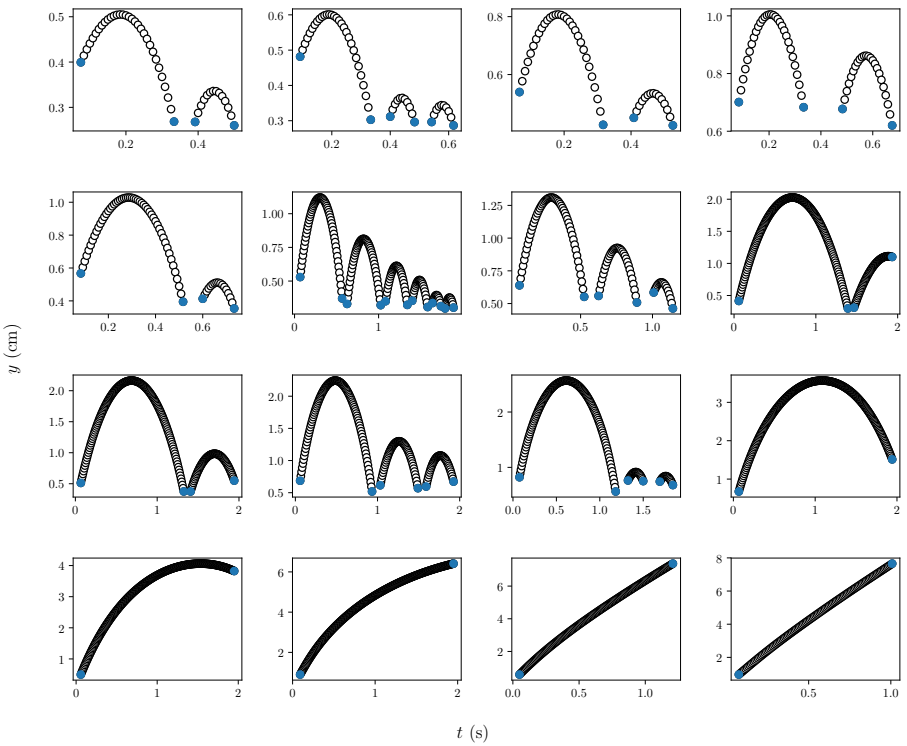


Figure 5: A series of filtered droplet trajectories arranged by increasing apoapse. The blue dots represent either the beginning and end of the experiment, or points at which the droplet is either coming into, or leaving contact with the surface.

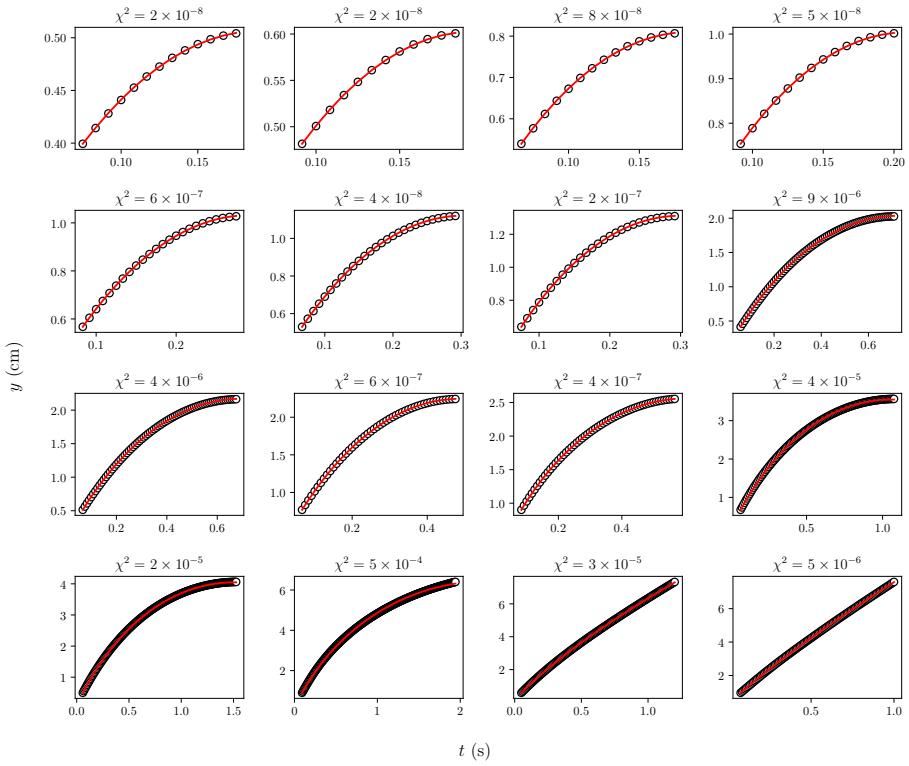


Figure 6: A series of droplet trajectories showing the results of the parameter estimation. The trajectories are shown only up to the apoapse of the first bounce. The red lines show the ODE solution with given the MLE parameter vector. $\chi^2$ goodness-of-fit varies between $1 \times 10^{-5}$ and $1 \times 10^{-8}$ with the better fit occurring typically for the droplets with the lowest apoapses.