# Chapter 1

# A Two Level $L_1$ Parameter Cascade Using the MM Alogrithm.

In the previous chapter, Brent's method was introduced and was used to tackle Parameter Cascade problems where the middle level uses a loss function which is difficult or to differentiate or simply has no well-defined derivative everywhere. It was remarked that Brent's method ensures that the different elements of the Parameter Cascade tend to be loosely coupled from each other and this allows one to combine different fitting methodologies for different levels with straightforwardly.

Here these ideas are developed futher. First, the $L_2$ based penalised fitting method is extended to the $L_1$ case. This new method is then used alongside Brent's method to implement a two level Paremeter Cascade with $L_1$ loss functions at both levels.

## 1.1 $L_1$ Estimation for the Inner Problem

Brent's method is designed to optimise real-valued functions over a real interval. In the previous it was extended to functions that take more than one real argument by optimising over each coordinate seperately when the Cauchy likelihood was optimised. However, there is no guarantee that this approach will perform well, and it can even fail entirely for functions that have an exotic topography or multiple local optima arranged unusually[1]. Even in the best case, optimising over each coordinate generates its own optimisation subproblem, which has the cumlative effect of increasing the running time of the alogrithm. Brent's method further requires the specification of a bounding box that contains the optimal point since it uses bisection, and that is harder and harder to do as the number of dimensions increases. All of these considerations mean that Brent's Method is highly unsuitable for peforming $L_1$ fitting over a space of functions which tend to have a large number of dimensions - by definition, there is one dimension introduced for each basis function used. Likewise, the non-differentiability of the absloute value function means that other approaches that implicitly rely on differentiability such as parabolic interpolation are inadvisable.

Instead a different approach will be employed, a generalisation of the Iteratively Reweighted Least Squares algorithm for computing the $L_1$ median of a set of $N$ items $\{x_1, \ldots, x_N\}$ to which an $L_1$ norm can be associated. The $L_1$ median is defined as the object $x$ that minimises $\sum_{i=1}^{N} |x - x_i|$. We will start by describing how IWLS can be used to compute the $L_1$ median of a set of real numbers. We will further show that this as an example of what is known as an MM alogrithm, and then proceed to

[1] Consider for example the problem of finding the minimum of the function $f(x) = x\sin(x)$ over the interval $[0, 13]$. It is easy to see that the minimum is not on the boundary points of the interval because $f(0) = 0, f(6) = -1.67$, and $f(13) = 5.45$. Brent's method fails to find the minimum. It claims the optimal value is given by $f(4.9) = -4.81$ though $f(11) = -10.99$.

straightforwardly extend this MM algorithm to produce a modified Penalised Sum of Squares problem that can be iteratively solved and reweighted to find the function that minimises a penalised $L_1$ norm.

### 1.1.1 An MM Algorithm For Computing the Median

Suppose that given a set of numbers $\{x_1, \ldots, x_N\}$ , one wished to find the number $x$ that minimised the $L_1$ distance between them:

$$SAE(x) = \sum_{i=1}^{N} |x_i - x|$$

It is well known that $SAE(x)$ is minimised by the sample median of the numbers [25][2]. The usual approach to computing the sample median - sorting the numbers and taking the one in the middle - can't be generalised to FDA problems, so we will use a different approach. The main difficulty is that the function $SAE(x)$ is not everywhere differentiable, which means that the usual derivative-based techniques such as gradient descent or Newton's method can't work. Instead an approach known as Majorise-Minimise or the MM Alogrithm will be used[9, 14, 15]. For a given iterate $x_n$, a function $M(x|x_n)$ is required with the following properties:

$$\begin{aligned} M(x|x_n) &\geq SAE(x) \\ M(x_n|x_n) &= SAE(x_n) \end{aligned}$$

The function $M(x|x_n)$ is said to majorise $SAE(x)$. The next iterate $x_{n+1}$ is then found as the value of $x$ that minimises $M(x|x_n)$. Thus:

$$\begin{aligned} SAE(x_{n+1}) &\leq M(x_{n+1}|x_n) \\ &\leq M(x_n|x_{n+1}) \\ &= SAE(x_n) \end{aligned}$$

If such a function $M(x|y)$ could be determined such that $M(x|y)$ would be straightforward to minimise, it is then possible to easily produce a sequence of iterates $x_n$ such that $SAE(x_{n+1}) \leq SAE(x_n)$ for all $n$. This pattern of monotone improvement in the objective function is similar to the EM Alogrithm. In fact, the EM algorithm is a special case of the MM algorithm[30] [3].

The most important question associated with the MM alogrithm is the construction of the majorising function because this tends to take up the bulk of the effort. Once the majoriser has been found, the algorithm is generally straightforward to implement, as will be seen shortly [16, 9]. Verifying a potential majoriser is usually straightforward, finding one in the first place is more difficult.The EM algorithm for example takes advantage of the probablistic structure of the problem and Jensen's inequality [4]. For an $L_1$ problem, the usual approach is to employ the Arithmetic Mean-Geometric Mean inequality [16]. Only the AM-GM inequality in its simplest form is required here, that the geometric mean of two numbers is less than or equal to their arithemtic mean:

[2]This is asserted without proof in [25], probably because the proof tends to be simultaneously awkward but trivial to those familiar with it. The amount of work required to demonstrate that the sample median minimises $SAE(x)$ is greatly reduced if one notes that $SAE(x)$ is a convex function in $x$. Any local minimum of a convex function is also a global minimum[1], so one only needs to show that for any sufficiently small $\epsilon$ that $SAE(\bar{x}) \leq SAE(\bar{x} \pm \epsilon)$, where $\bar{x}$ is the sample median.

[3]When applied to maximisation problems, MM instead stands for Minorise-Maximise. This case is the same except the surrogate function is required to be less than or equal to the objective function and it is maximised on each iteration. Thus, each iteration drives the objective function upwards.

[4]The EM alogrithm is intended to maximise the log-likelihood and drives it upwards on each iteration, so it's an example of a Minorise-Maximise algorithm.

$$\sqrt{xy} \leq \frac{x+y}{2}$$

It's worth noting that the AM-GM inequality is in fact a special case of Jensen's Inequality since the log function is concave:

$$\begin{aligned}\log(\frac{x+y}{2}) &\geq \frac{\log x}{2} + \frac{\log y}{2} \\ &= \log\sqrt{x} + \log\sqrt{y} \\ &= \log\sqrt{xy}\end{aligned}$$

It is possible to exploit the AM-GM inequality to majorise an $L_1$ regression problem by a weighted $L_2$ problem. One can represent the $L_1$ norm as a geometric mean, which then allows for the $L_1$ norm to majorised and seperated by a weighted sum of squares. Given an interate $x_n$, the AM-GM inequality implies that:

$$\begin{aligned}|y-x| &= \sqrt{(y-x)^2} \\ &= \sqrt{\frac{(y-x)^2}{|y-x_n|}|y-x_n|} \\ &\leq \frac{1}{2}\left(\frac{(y-x)^2}{|y-x_n|} + |y-x_n|\right)\end{aligned}$$

This in turn implies that:

$$\begin{aligned}\sum|x_i-x| &\leq \frac{1}{2}\sum\left(\frac{(x_i-x)^2}{|x_i-x_n|} + |x_i-x_n|\right) \\ &= \frac{1}{2}\sum\frac{(x_i-x)^2}{|x_i-x_n|} + \frac{1}{2}\sum\left(|x_i-x_n|\right)\end{aligned}$$

The $L_1$ problem is thus majorised by a weighted least squares problem. The $\frac{1}{2}\sum|x_i-x_n|$ term is constant with respect to $x$, so neglecting it makes no difference to the choice of $x$ that is optimal. Likewise, multiplying the weighted least squares problem by a positive constant doesn't change the optimal value either, so the $\frac{1}{2}$ term can be eliminated by multiplying by 2. The optimal value $x_{n+1}$ can thus be found by minimising this weighted least squares score:

$$\sum\frac{(x_i-x)^2}{|x_i-x_n|}$$

The alogrithm thus consists of finding the value of $x$ that minimises the least squares error inversely weighted by the residuals from the previous iteration.

### 1.1.2 Penalised $L_1$ Fitting

For the case of penalised regression, the penalised sum of absolute errors is defined by:

$$PENSAE(f|\theta,\lambda) = \sum|x_i - f(t_i)| + \lambda\int|Tf|^2dt$$

Here $T$ is used instead of $L$ to denote a differential operator that might not necessarily be linear.[5] As before, this can be majorised by a weighted sum of a residual-weighted penalised sum of squared

[5] In some situations $T$ could even be an integral operator. This could easily be the case for example if the observed values were the measured velocities of a vehicle, and the penalty was intended to impose constraints on quantities such as the distance travelled or fuel consumed

errors, and a vestigal $\sum |x_i - f_n(t_i)|$ term that is only included for completeness and can be safely ignored in the course of the actual optimisation.

$$PENSAE(f) \leq \frac{1}{2} WPENSSE(f|f_n, \theta, 2\lambda) + \frac{1}{2}\left(\sum |x_i - f_n(t_i)|\right) \tag{1.1}$$

$$= \frac{1}{2}\left(\sum \frac{[x_i - f(t_i)]^2}{|x_i - f_n(t_i)|} + 2\lambda \int |Tf|^2 dt\right) + \frac{1}{2}\left(\sum |x_i - f_n(t_i)|\right) \tag{1.2}$$

To find the function that minimises the penalised $L_1$ error, one repeatedly finds the function that minimises $WPENSSE$ with the previous set of residuals used as inverse weights. This produces a sequence of fitted functions for which the penalised sum of absolute errors is montonically forced downwards.

### 1.1.3 Discussion

The sequence of penalised errors $PENSAE(f_n)$ is monontone decreasing but cannot be less than zero, so it is a bounded monotone sequence. The Monotone Convergence Theorem for sequences of real numbers[24] thus guarantees that a given generated sequence $PENSAE(f_n)$ will always converge to a limit. There are two cavaets. First, the sequence might converge to a different point depending on the starting values - there is no guarantee that the sequence will converge to the lowest possible value of $PENSAE$. Second, there is no guarantee that the underlying sequence of functions will converge, and may just oscillate between several points. The sequence $-1, 1, -1, \ldots$ does not converge but the associated sequence of absolute values $1, 1, 1, \ldots$ does.

This approach of associating the objective function with more standard problem that acts as a surrogate is employed in the literature on the EM Algorithm. For example, in the introductory chapter of [19], the authors discuss how a multinomial estimation problem can be transformed into a binomial problem with missing data by artificially splitting one of the cells; they then construct a simple iterative EM scheme that can then be repeatedly iterated to estimate parameters for the original multinomial. They even remark that the the surrogate problems associated with EM alogrithms tend to be easy to solve using existing tools in the field. Likewise, the $L_1$ problem has been replaced here with a surrogate sequence of weighted $L_2$ problems that can easily solved using the `FDA` package. Since the `FDA` package does much of the heavy lifting, the actual code for implementing penalised $L_1$ regression is brief.

The literature on the MM algorithm remarks that it is simple to implement and good at tackling high dimesional penalised regression, though convergence can be slow [30]. These claims are borne out when the convergence of the method is examined in Section 1.1.4 below.

#### Estimating the Worst Case Running Time

The arguments used to prove the Monotone Convergence Theorem can be extended a little to give a crude and weak estimate of how long the MM algorithm runs before the associated $PENSAE(f_n)$ sequence convergences to within a given tolerance. Suppose there were a monotone decreasing sequence bounded between $a$ and $b$ so that $a \geq x_0 \geq x_1 \geq \cdots \geq x_n \geq \cdots \geq b$. By the Monotone Convergence Theorem, this sequence converges to some value $x^*$ such that $a \geq x^* \geq b$ and $x_n \geq x^*$ for all $n$.

Given $\epsilon > 0$, it is not hard to see that there exists an associated number $N(\epsilon)$ such that there exists an $n \leq N(\epsilon)$ with the property that $|x_{n+1} - x_n| \leq \epsilon$, where $N(\epsilon)$ is defined by $N(\epsilon) = \lceil (a-b)/\epsilon \rceil + 1$.[6] This result gives a crude and weak bound on how long it takes an before algorithm with a descent property achieves a tolerance threshold. If for example, the MM algorithm is programmed to terminate as soon as $|PENSAE(f_{n+1}) - PENSAE(f_n)| \leq 10^{-6}$, then one can expect the alogrithm to terminate within around $10^6$ iterations in the worst case.

[6] If $|x_{n+1} - x_n| > \epsilon, \forall n \leq N(\epsilon)$, it would be the case that $x_{N(\epsilon)} < b$, which contradicts the assumption that every element $x_n$ of the sequence is bounded below by $b$.

Such woefully slow performance is unusual, but not entirely unprecedented. An example of a simple Poisson estimation problem where the EM Algorithm exhibits sublinear $\mathcal{O}(1/n)$ convergence[7] is given in [15]. The Implicit Filtering fitting algorithm previously discussed is another example of an incredibly slow method, requiring around 200 iterates before the sequence of values objective function converges to within $10^{-4}$ of each other [8].

This result is very pessimistic though, the EM and MM Alogrithms usuallly converge linearly [19, 9] The MM alogrithm for $L_1$ fitting does not quite converge linearly though as shall be seen in Section 1.1.4 below.

[7] A possibly vector-valued sequence $\{\mathbf{x}_n\}_{n=1}^{\infty}$ is said to exhibit $\mathcal{O}(1/n)$ convergence or converages at rate $\mathcal{O}(1/n)$ if there exists constants $c_1 > c_2 > 0$ and a limit value $\mathbf{x}^*$ such that $c_1/n \geq \|\mathbf{x}_n - \mathbf{x}^*\| \geq c_2/n$.

[8] The exact order of convergence of Implicit Filtering is difficult to determine. Experimentation suggests that the rate of convergence is sublinear.

### 1.1.4 Testing the Algorithm on the Melanoma Data

Since minimising $PENSAE$ is a problem over many dimesions, plotting the objective function to verify that the optimal function has been found isn't possible. Instead the MM alogrithm described in Section 1.1.2 will be tested by applying it to the melanoma data pertubred by random noise. Further, the covergence of the algorithm for the original melanoma dataset will be examined.

Figure 1.1 presents the $L_1$ and $L_2$ inner fits to the melanoma data corrupted by Cauchy distributed noise. The value of $\omega$ is held fixed at the reasonable value of 0.3, which was chosen as being roughly the average of the two different estimates of $\omega$ from computed in the previous chapter. It is apparent from the Figure 1.1 that the MM fit is robust against outliers, tends to ignore more deviant points, and even manages to remain similar to the original fit. The least-squares fit tends to chase the heavy-tailed noise on the other hand. This is strong evidence that the curve that minimses $PENSAE$ has been found and that the method has been implemented correctly.

Figures 1.2 and 1.4 plot the convergence of $SAE$ and $PENSAE$ over the course of the alogrithm. Note that the PENSAE statistic doesn't quite actually converge monotonically as the theoretical analysis predicted. Instead, it fluctuates before settling down to the typical and expected pattern of languid monotone decline. Upon investigation, it was determined that over the first handful of iterations the range of the weights applied to the observations on each iteration, that were computed using the residuals from the previous iteration, grew very rapidly. By the fourth iteration, the lowest weight is equal to 1.48, and the highest was equal to $4.8 \times 10^6$. It seems that this rapid and large change produces qualititative changes in behaviour before the algorithm manges to 'burn in'. It is likely that observations with low weights are being effectively censored after a few iterations due to roundoff error. It was found that imposing a minimum threshold for the weights by adding a constant to all the residuals before proceeding to computing the weights smooths out this behaviour, but doesn't eliminate it entirely.

Figure 1.3 plots the convergence of the coefficient vectors $\mathbf{c}_n$. This log-plot suggests that the sequence of fitted coefficient vectors $\mathbf{c}_n$ converges linearly since $\|\mathbf{c}_{n+1} - \mathbf{c}_n\| \approx C\|\mathbf{c}_n - \mathbf{c}_{n-1}\|$ as $n \to \infty$.

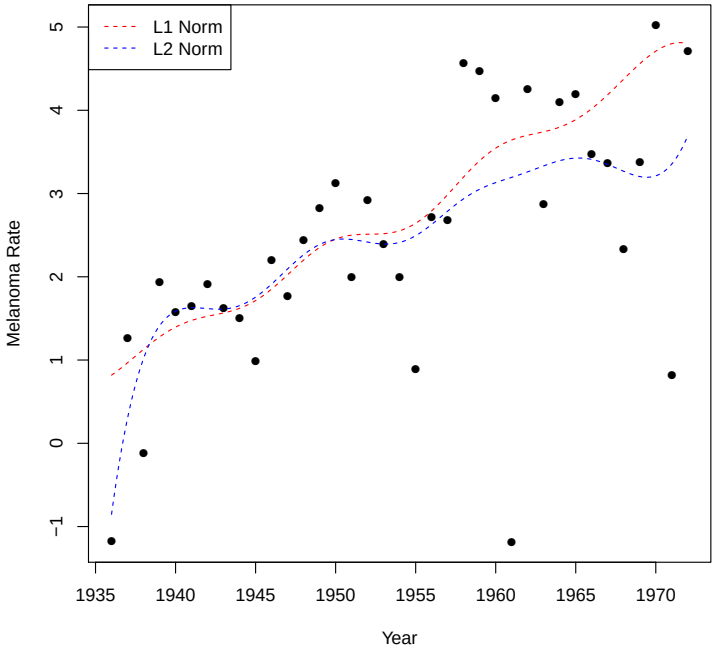


Figure 1.1: Comparison of $L_1$ and $L_2$ inner fits to Cauchy perturbed data with $\omega$ fixed at 0.3

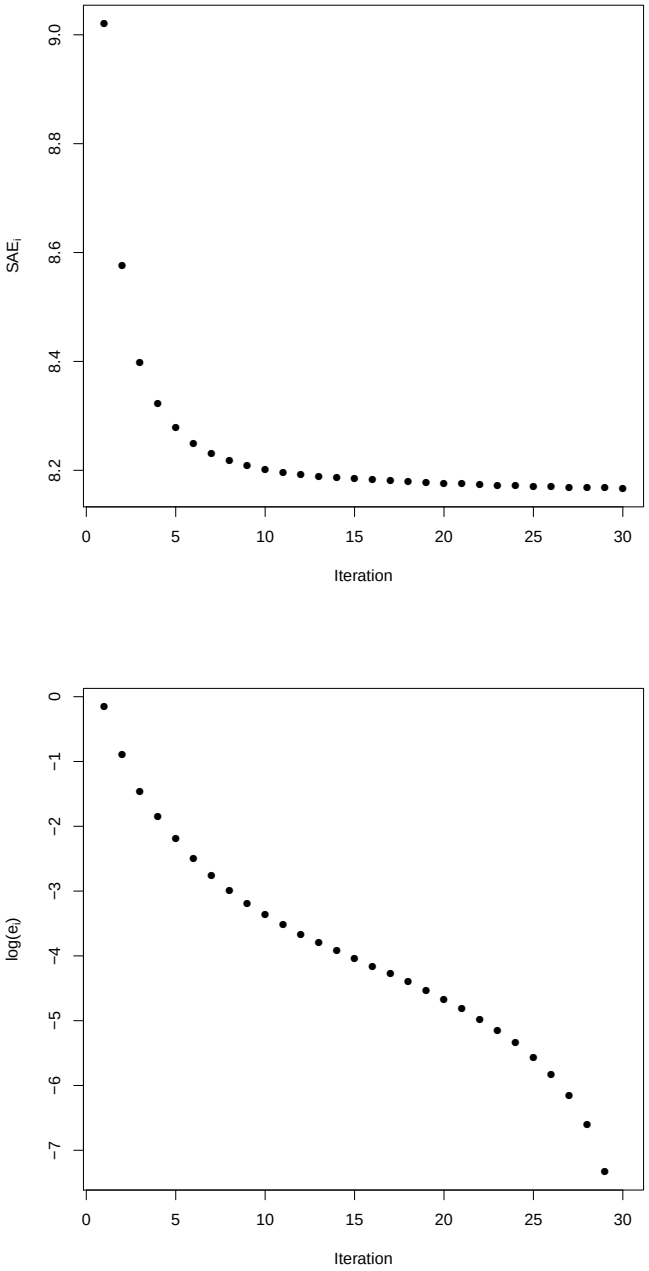


Figure 1.2: Plot of values and log differences for SAE Statistic

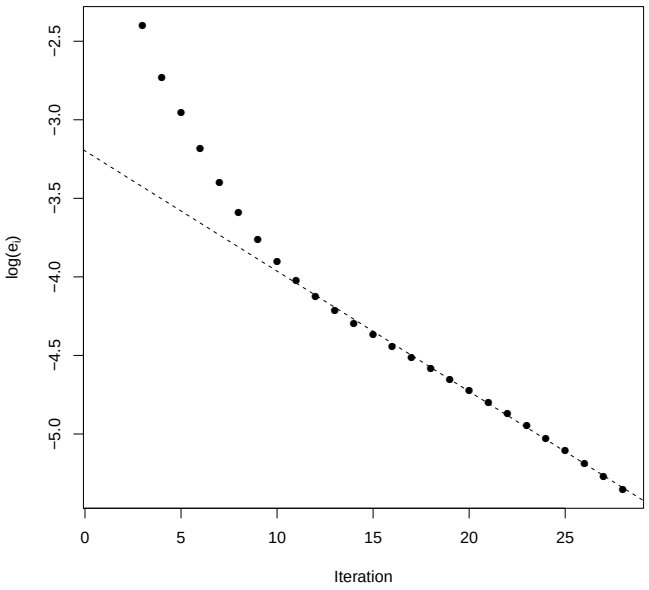


Figure 1.3: Plot of log norm differences for coefficients. Note that they tend to settle on a line.

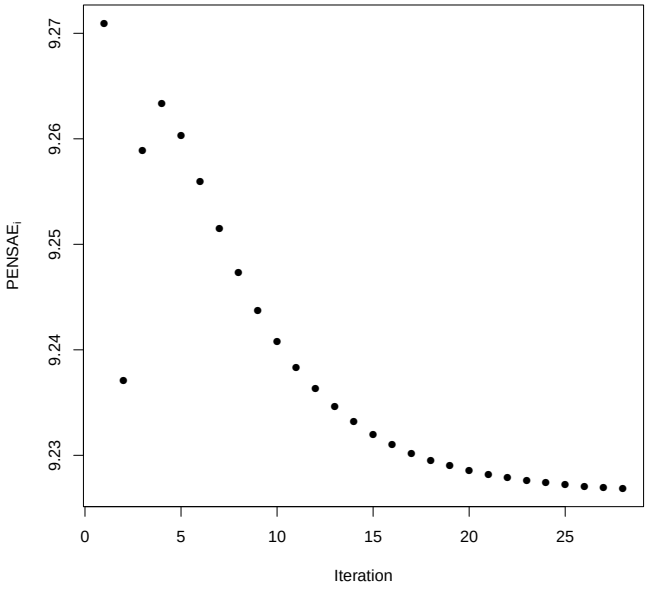


Figure 1.4: Plot of PENSAE statistic as the algorithm proceeds.

## 1.2 The Two Level Parameter Cascade with $L_1$ Norm

The inner problem of the parameter cascade is a semiparametric least squares regression model. The fitted function is modeled as a weighted sum of a solution to the differential equation (parametric), and a non-parametric residual. The lambda term governs how big the residual is allowed to be relative to the the least squares error term

If the usual least-squares error function is used, the inner problem will probably struggle with outliers and heavy tailed errors as is the case for any form of least-squares regression.

For high order differential operators like that used to model the melanoma data, there are many degrees of freedom associated with the differential operator's solution set. The omega and lambda parameters don't strongly constrain the lower level of the cascade. There is thus little capacity for the higher levels of the cascade to restrain the lowest level through altering the lambda and omega parameters and the parameter cascade must use robust estimation at every level.

In the previous chapter, it was discussed how Brent's Method can be used to tackle the middle problem without derivatives and then used this approach to optimise a highly irregular loss function. In the previous section, the MM algorithm was used to optimise the inner problem with an $L_1$ norm.

Combining the two methods, it is very straightforward to implement a two-level parameter cascade with $L_1$ errors at both levels.

In Figure 1.5, the result of fitting a two level $L_1$ Parameter Cascade with L1 errors is plotted. It can be seen that the $SAE(\omega)$ function is irregularly shaped. In Figure 1.6 both the $L_1$ and $L_2$ fits to Cauchy-perturbed melanoma data are shown. Figure 1.7 plots the results of applying the $L_1$ and $L_2$ Parameter Cascades to the original and perturbed Melaonoma data, alongside mixed versions where the $L_1$ loss function is used for the inner fitting and $L_2$ loss function for the middle fitting and vice versa.

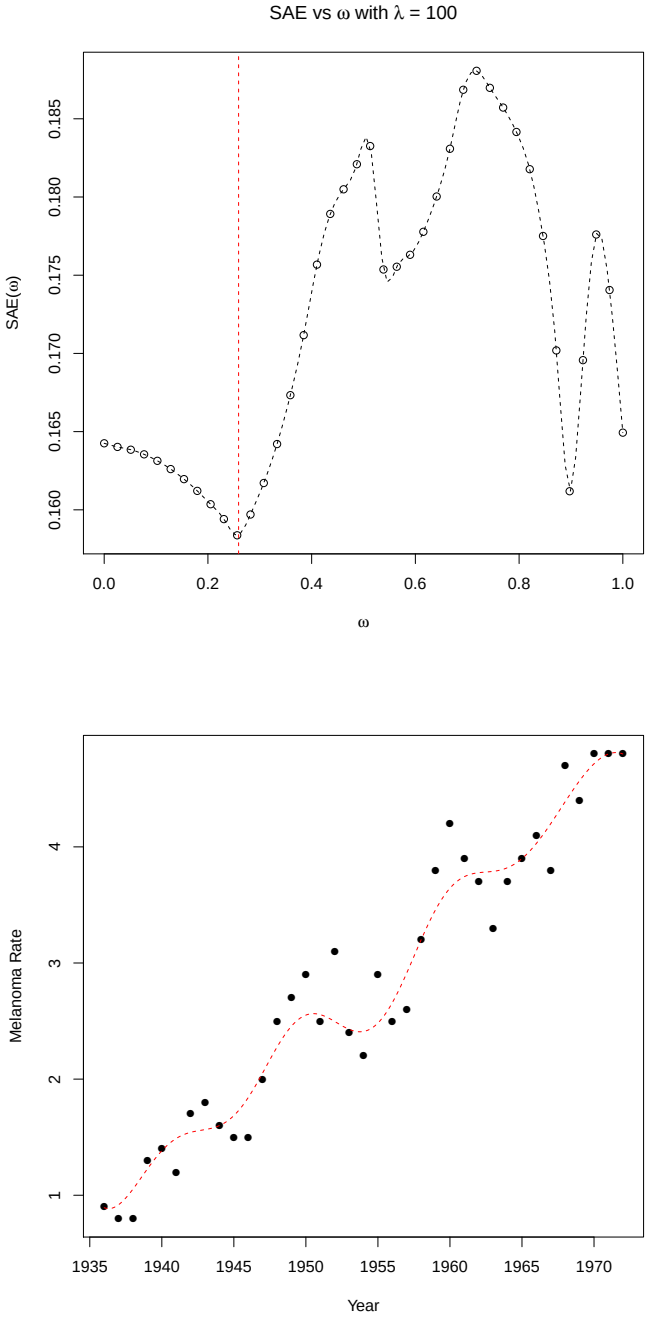


Figure 1.5: Fitting an $L_1$ Parameter Cascade to the Melanoma Data

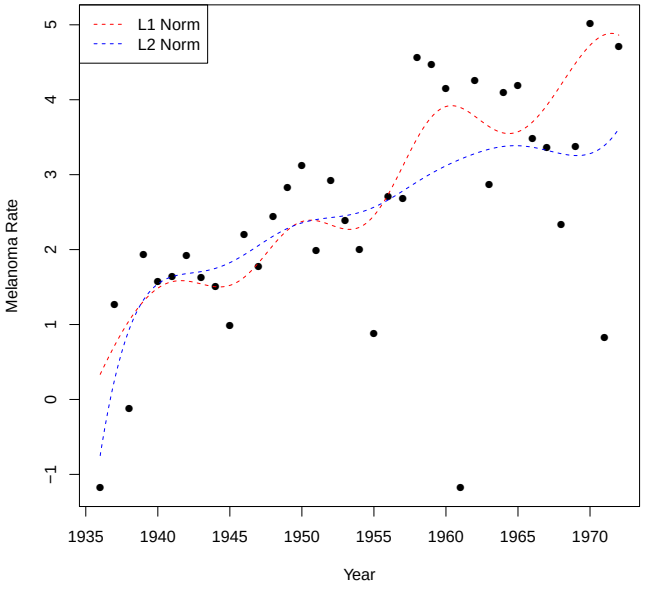


Figure 1.6: $L_1$ and $L_2$ Parameter Cascades with the same perturbed data as in Figure 1.1. Compare the $L_1$ curve in this plot with the one in Figure 1.5.

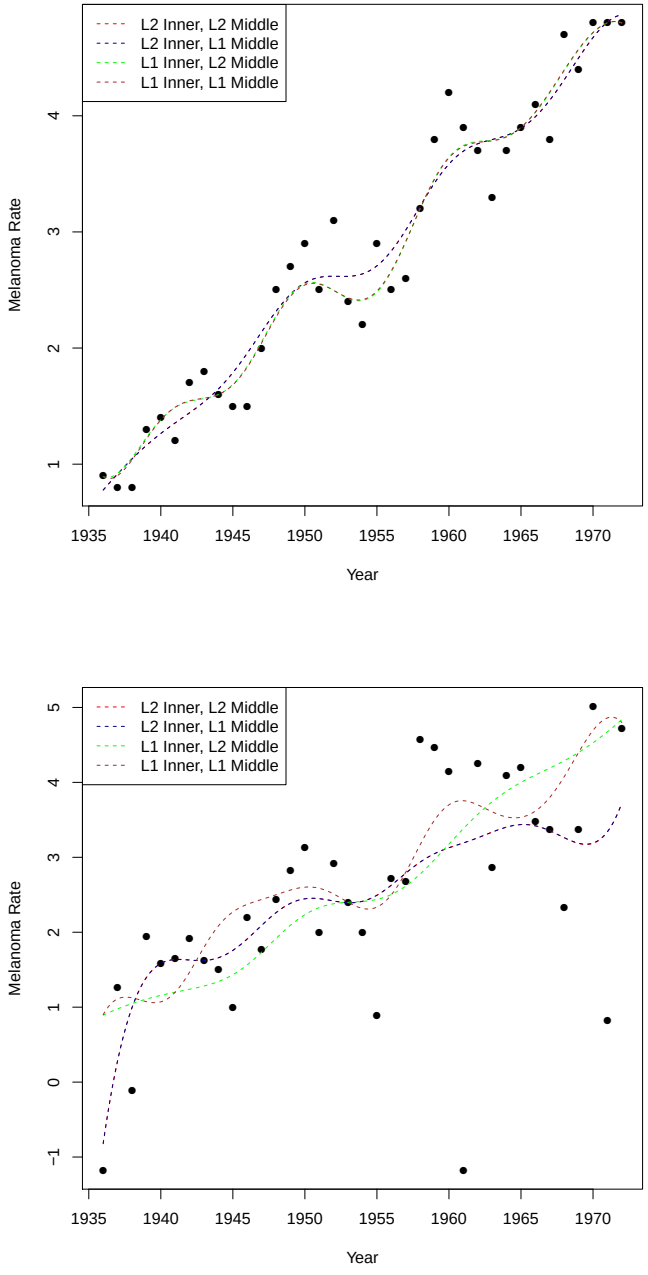


Figure 1.7: All possible combinations of $L_1$ and $L_2$ loss functions that can be used for the Parameter Cascade. The top plot applies them to the orginal melanoma data, the bottom to the same perturbed data as in Figures 1.1 and 1.6

## 1.3 Acclerating The Rate of Convergence

The MM Alogrithm is very sluggish, and this is a well known weakness of both itself and the EM algorithm. The literature however suggest that this problem could be easily ameliorated in this particular case because of a special feature present. In practice one doesn't want to fit the full model, but only wants to compute an associated summary statistic that determines how good a given choice of parameters is. It will often be the case that only the value of $PENSSAE$ or $GCV$ or $SAE$ associated with a given choice of parameters is required as inputs to an optimisation routine, and it is not desireable to iterate until the full model converges if this effort can be avoided.

MacLanan and Krishnan discuss the situation where one only wants to compare the likelihoods between a restricted model and a full model. They suggest the use of sequence acceleration methods to rapidly extract the likelihoods [19] instead of running the EM algorithm to completion since the full models aren't needed. The literature on the MM algorithm claims that acceleration methods for the EM algorithm translate quite easily to the MM case [30]. On this basis, we explored whether this approach might be applied here.

The textbook approach employed is known as Aitken Acceleration[6, 19]. Suppose that there is a sequence $x_0, x_1, x_3, \ldots$ converging to a limit $x^*$. Aitken's method makes the ansatz that $x_{n+1} - x^* \approx C(x_n - x^*)$ for some constant $C$. Many iterative algorithms in statistics exhibit this pattern as discussed in Section 1.1.3. This suggests the following equation:

$$\frac{x_{n+1} - x^*}{x_n - x^*} \approx \frac{x_n - x^*}{x_{n-1} - x^*}$$

Solving for $x^*$ gives the accelerted sequence.

There is an equivalent definition that is easier to generalise [10]. Consider a sequence defined by functional iteration so that $x_{n+1} = F(x_n)$ for some function $F(\cdot)$ Define the error sequence by $e_n = x_{n+1} - x_n = F(x_n) - x_n$. The function $g(x) = F(x) - x$ returns the error associated with any value, and the limit of the sequence satisfies $g(x^*) = 0$. Suppose one knew the inverse of $g(x)$, which will be denoted by $h(e)$. Then $x^*$ could be found by evaluating $f(0)$. The next best thing would be to use the values of the sequence to approximate $h(e)$, and then evaluate this approximate function at zero instead. The Aitken method approximates $h(e)$ by linear interpolation between $(e_n, x_n)$ and $(e_{n-1}, x_{n-1})$, and then evaluates this approximation at $e = 0$.

### 1.3.1 Illustrative Example: Using Imputation to Fit an ANOVA Model With Missing Data

For illustrative purposes, we will make use of an example from chapter 2 of [19]. The authors discuss fitting an ANOVA model to a factorial experiment where some of the values are missing. They proceed by using the fitted model to estimate the missing values; fitting the model again with the new imputed values; and using the new fitted values in turn to again update the estimates of missing values. The process is repeated until convergence. In the text, the authors do not work here with likelihood or any probablistic models and treat the question as purely a regression problem. This is similar to our $L_1$ fitting problem.

The authors' example was implemented again in R.[9]. For each iteration, the $SSE$ statistic was computed. This defines an associated sequence $\{SSE_1, SSE_2, \ldots, SSE_n, \ldots\}$. Applying Aitken's method to this sequence produces a new sequence $\{ASSE_n\}$. As can be seen in Figure 1.8 and Table 1.1, the accelerated sequence converges far more quickly to the limit of the $\{SSE_i\}$ sequence than the original sequence.

### 1.3.2 Generalisations

Outside of more specialised texts, Statistics is generally content with Aitken's method and multivariate generalisations. Exploring more powerful methods can justified in two circumstances. The first is that if one is running the algorithm over and over again such that an increase in speed over many iterations means the effort invested is worth it. This might be the case for example if one wanted to use the bootstrap to model the distribution of an likelihood ratio statistic computed using the EM Algorithm as previously described. The second is if the sequence is difficult to accelerate. In this situation, it shall be seen that both conditions apply.

As a field of study, sequence acceleration is closely related to time series analysis. A generic first order autoregressive model is given by:

$$x_{n+1} = f(x_n, n) + \epsilon_n$$

Consider the case where there are both no random errors so that $\epsilon_n$ is always zero, and the sequence converges to a limit. Here, the problem of deterimining the long term value of the sequence from a set of observations is equivalent to that of accelerating the sequence. If the specific form of $f(x_n, n)$ is known, there can often be a specific acceleration method that can exactly extract the limit. For illustration, suppose there were a sequence of the following form, but the parameters $\beta_0$ and $\beta_1$ were unknown:

$$x_n = \beta_0 + \frac{\beta_1}{n} \tag{1.3}$$

As $n$ goes to infinity, $x_n$ converges to $\beta_0$. It is not difficult to show that the limit $\beta_0$ can be found by applying the following sequence transformation:

$$\begin{cases} \hat{\beta}_{1,n} = \dfrac{x_n - x_{n+1}}{\left(\frac{1}{n} - \frac{1}{n+1}\right)} \\ \\ \tilde{x}_n = x_n - \dfrac{\hat{\beta}_{1,n}}{n} \end{cases} \tag{1.4}$$

If the transformation (1.4) is applied to a sequence of values $x_1, x_2, \ldots, x_n, \ldots$ that is of form (1.3), then the transformed sequence $\tilde{x}_1, \tilde{x}_2, \ldots, \tilde{x}_n, \ldots$ will have the property that $\tilde{x}_n = \beta_0$ for all

[9] The $SSE$ statistic here is different from the $RSS$ statistic presented in the text this example was taken from. The new code converges to the same estimates as in the original, so the example has been re-implemented correctly. It was not possible to determine with what degrees of freedom $RSS$ was associated with.

$n$. Likewise, the Aitken method is exact for sequences of the form $x_{n+1} = \beta_0 + \beta_1 x_n$, and so can be thought of as the deterministic analogue of an $AR(1)$ model.

The process of acceleration isn't quite so neat in practice because sequences don't adhere perfectly to these simple forms. Instead, the best that can be realistically hoped for is that the transformed sequence converges to the same limit as the original, but the rate of convergence is higher. For example, if transformation (1.4) is applied to a sequence of the form $y_n = \beta_0 + \frac{\beta_1}{n} + \frac{\beta_2}{n^2}$, then the transformed sequence is now of the form $\tilde{y}_n = \beta_0 + \mathcal{O}(\frac{1}{n^2})$, which converges to $\beta_0$ more quickly than the original sequence.[10]

Suppose a convergent sequence is of the form $x_{n+1} = f(x_n)$ with $f(\cdot)$ differentiable and $x^*$ is the limit. Using a first order Taylor expansion, it can be seen that for sufficiently large $n$, $x_{n+1} \approx x^* + f'(x^*)(x_n - x^*)$. In this case, Aitken acceleration has a decent chance of accelerating the sequence so long as it has 'burned in' sufficiently.

One generalisation, proposed in [10] is to use higher order polynomials to model the inverse error function $h(e)$. So $h(e)$ would be approximated by a quadratic through $(e_n, x_n), (e_{n-1}, x_{n-1})$ and $(e_{n-2}, x_{n-2})$. Making $e$ the independent variable here instead of $x$ means the estimated limit can simply be found by evaluating the appproximating quadratic at $e = 0$ instead of having to find the correct root of a quadratic to compute each element of the accelerated sequnce.

Another approach is to simply apply Aitken Acceleration to the sequence twice.

Both these approaches were attempted for the missing data model, and the results can be seen in Table 1.1 and Figure 1.9. It can be seen that both methods improve convergence, though double Aitken acceleration is more effective (and easier to implement).

One can take the process further. For the missing values linear model, these higher-order methods converge very rapidly and are prone numerically instability thereafter due to the error terms being so small, so plotting or tabluating them was not worth the additional clutter. If the Aitken method is applied three times to the original sequence, the first entry yields the limit immediately and there is no need to go any further. Applying the quadratic method twice in a row produces a new sequence for which the first entry is within $10^{-12}$ of the limit.

**Other Approaches:** There are alternative approaches besides those described here. For example, the EM and MM alogrithms generate a sequence of coeffcient vectors $\{\mathbf{c}_0, \mathbf{c}_1, \ldots, \mathbf{c}_n, \ldots\}$ with $\mathbf{c}_{n+1} = \mathbf{F}(\mathbf{c}_n)$ for some function $\mathbf{F}(\cdot)$. In our particular situation, the function $\mathbf{F}(\cdot)$ would denote the operator that takes a coefficient vector and returns the coefficient vector that minimises the associated $WPENSSE$ problem (1.1). The limit of this sequence - should it exist - is a solution to the equation $\mathbf{c} = \mathbf{F}(\mathbf{c})$. It is proposed in the literature to use Newton or Quasi-Newton methods such as those described in the chapter on Brent's method to numerically solve this fixed point equation [4, 6]. The idea is that such methods will find the fixed point more rapidly than simply iterating $\mathbf{F}(\cdot)$ until one gets sufficiently close to the limit. These methods have the disadvantage of being more complex and time consuming to implement than the univariate acceleration methods.

[10] Doing the algebra, it can be seen that it is now the case that $\hat{\beta}_{1,n} = \beta_1 + \beta_2 \left[\frac{2n+1}{n(n+1)}\right]$, and so $\tilde{y}_n = \beta_0 + \beta_2 \left[-\frac{2n+1}{n^2(n+1)} + \frac{1}{n^2}\right] = \beta_0 + \beta_2 \left[\frac{-n^3-n^2+n}{n^4(n+1)}\right]$.

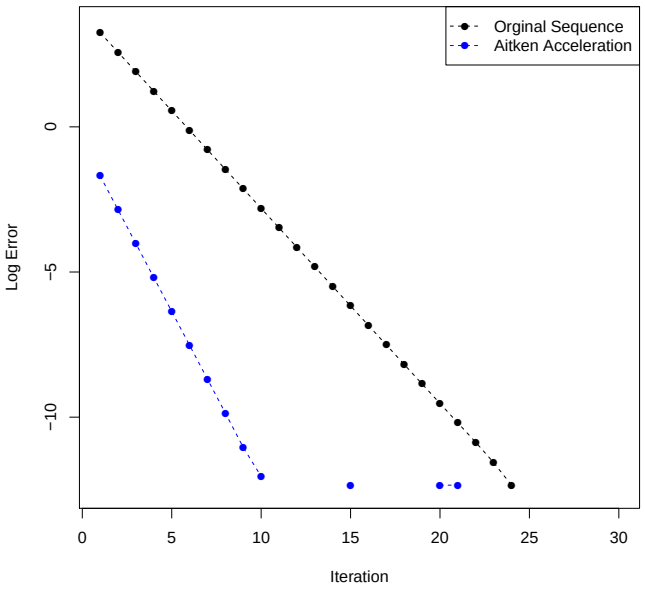


Figure 1.8: Log Errors for original sequnce of SSE values and the accelerated one

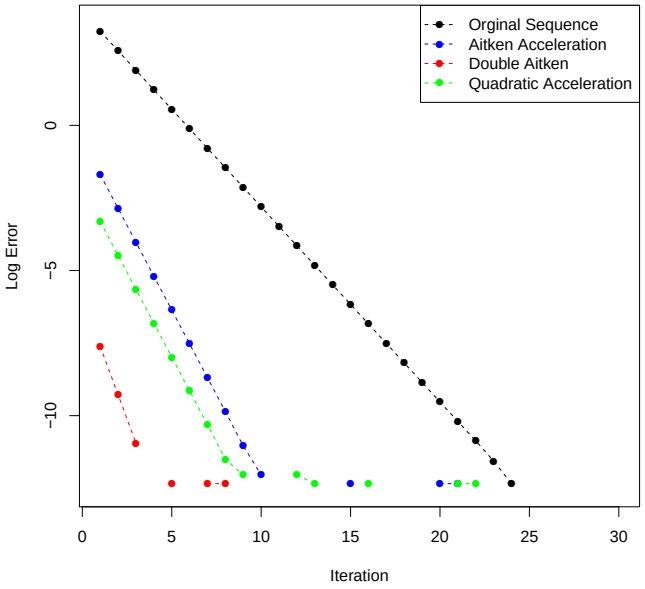


Figure 1.9: More sophisticated acceleration methods can provide a further boost to convergence. There are gaps in the plot because the more accelerated iterations have no valid log error since R cannot numerically distinguish them from the final limit.

Table 1.1: Iterations of the original sequence $SSE_n$, the accelerated sequence $ASSE_n$, the quadratically accelerated sequence $QASSE_n$, and the doubly accelerated sequence $DASSE_n$.

| | $SSE_n$ | $ASSE_n$ | $QASSE_n$ | $DASSE_n$ |
|---|---|---|---|---|
| 1 | 4949.6944444444 | 3203.8032711619 | 3203.7834325738 | 3203.7829457122 |
| 2 | 3575.1658950617 | 3203.7843303225 | 3203.7829788622 | 3203.7829457359 |
| 3 | 3282.7945625667 | 3203.7830400346 | 3203.7829479917 | 3203.7829457364 |
| 4 | 3220.5935028609 | 3203.7829521582 | 3203.7829458900 | 3203.7829457364 |
| 5 | 3207.3596279977 | 3203.7829461738 | 3203.7829457469 | 3203.7829457364 |
| 6 | 3204.5439391293 | 3203.7829457662 | 3203.7829457371 | 3203.7829457364 |
| 7 | 3203.9448588925 | 3203.7829457385 | 3203.7829457365 | 3203.7829457364 |
| 8 | 3203.8173952920 | 3203.7829457366 | 3203.7829457364 | 3203.7829457364 |
| 9 | 3203.7902754193 | 3203.7829457364 | 3203.7829457364 | 3203.7829457364 |
| 10 | 3203.7845052414 | 3203.7829457364 | 3203.7829457364 | 3203.7829457364 |
| 11 | 3203.7832775456 | 3203.7829457364 | 3203.7829457364 | 3203.7829457364 |
| 12 | 3203.7830163340 | 3203.7829457364 | 3203.7829457364 | 3203.7829457364 |
| 13 | 3203.7829607572 | 3203.7829457364 | 3203.7829457364 | 3203.7829457364 |
| 14 | 3203.7829489323 | 3203.7829457364 | 3203.7829457364 | 3203.7829457364 |
| 15 | 3203.7829464164 | 3203.7829457364 | 3203.7829457364 | 3203.7829457364 |
| 16 | 3203.7829458811 | 3203.7829457364 | 3203.7829457364 | 3203.7829457364 |
| 17 | 3203.7829457672 | 3203.7829457364 | 3203.7829457364 | 3203.7829457364 |
| 18 | 3203.7829457430 | 3203.7829457364 | 3203.7829457364 | 3203.7829457364 |
| 19 | 3203.7829457378 | 3203.7829457364 | 3203.7829457364 | 3203.7829457364 |
| 20 | 3203.7829457367 | 3203.7829457364 | 3203.7829457364 | 3203.7829457364 |
| 21 | 3203.7829457365 | 3203.7829457364 | 3203.7829457364 | 3203.7829457364 |
| 22 | 3203.7829457364 | 3203.7829457364 | 3203.7829457364 | 3203.7829457364 |
| $\infty$ | 3203.7829457364 | 3203.7829457364 | 3203.7829457364 | 3203.7829457364 |

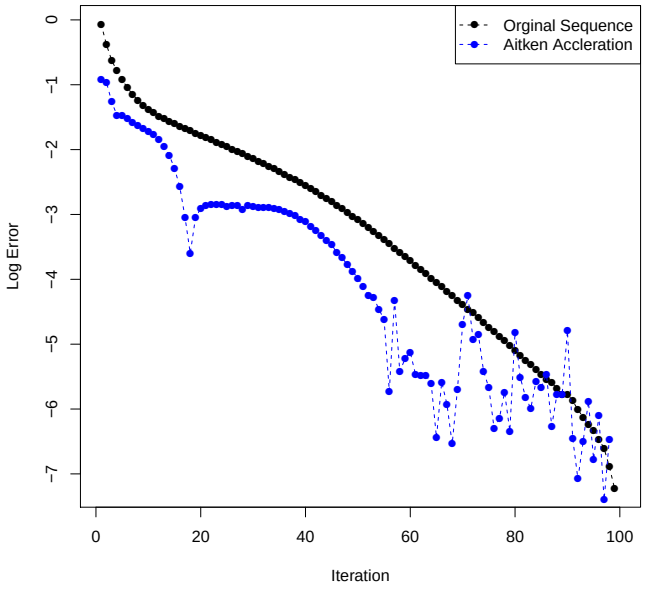


Figure 1.10: Accelerating the SAE sequence generating by the $L_1$ fitting alogrithm using Aitken's Method. The improvement in covergence is mediocre.

### 1.3.3 Accelerating the $L_1$ Fitting Alogrithm

The $L_1$ fitting algorithm is much more difficult to accelerate as can be seen in Figures 1.10 and 1.11. Even advanced acceleration methods recommended for slowly converging sequences that the Aitken method cannot tackle such as the Epsilon Algorithm[8, 21] or Lubkin's W transform [28, 21] yield much improvement[11]. The $SAE$ sequence is apparently either numerically ill-behaved or of a very unusual form.

To conclude, it is possible to save some time by acceleration, but the scope for doing so is limited and the process would have to be monitored carefully to ensure that numerical instability isn't causing trouble.

[11]Several other methods proposed in [21] were such as Levin transforms were also attempted. They proved similarly unsatisfactory

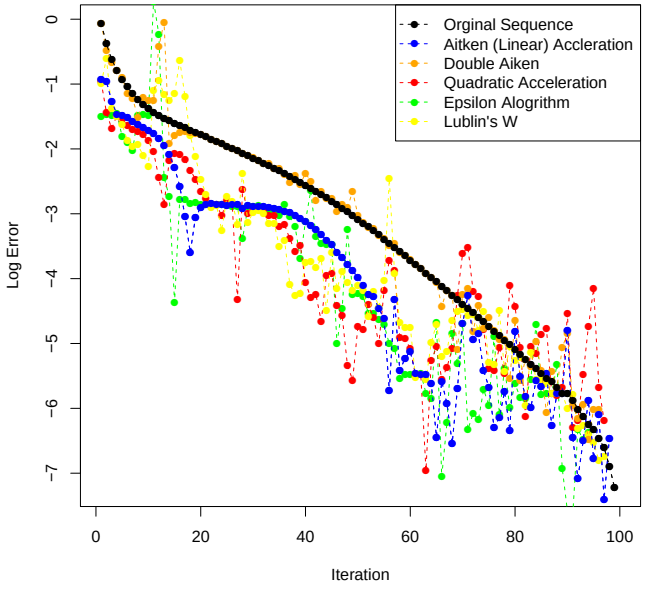


Figure 1.11: Accelerating the SAE sequence using multiple methods. Aitken's method performs the best, despite it's lack of sophistication.

# Bibliography


[1] Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[2] Richard P Brent. *Algorithms for minimization without derivatives*. Courier Corporation, 2013.

[3] Jiguo Cao and James O Ramsay. Parameter cascades and profiling in functional data analysis. *Computational Statistics*, 22(3):335–351, 2007.

[4] Kwun Chuen Gary Chan. Acceleration of expectation-maximization algorithm for length-biased right-censored data. *Lifetime data analysis*, 23(1):102–112, 2017.

[5] Edwin KP Chong and Stanislaw H Zak. *An introduction to optimization*, volume 76. John Wiley & Sons, 2013.

[6] Arthur P Dempster, Nan M Laird, and Donald B Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)*, 39(1):1–22, 1977.

[7] Bengt Fornberg. Generation of finite difference formulas on arbitrarily spaced grids. *Mathematics of computation*, 51(184):699–706, 1988.

[8] PR Graves-Morris, DE Roberts, and A Salam. The epsilon algorithm and related topics. *Journal of Computational and Applied Mathematics*, 122(1-2):51–80, 2000.

[9] David R Hunter and Kenneth Lange. A tutorial on MM algorithms. *The American Statistician*, 58(1):30–37, 2004.

[10] Eugene Isaacson and Herbert Bishop Keller. *Analysis of numerical methods*. Courier Corporation, 2012.

[11] Carl T Kelley. *Implicit filtering*, volume 23. SIAM, 2011.

[12] C.T. Kelley. A brief introduction to implicit filtering. `https://projects.ncsu.edu/crsc/reports/ftp/pdf/crsc-tr02-28.pdf`, 2002. [Online; accessed 12-October-2019].

[13] Jack Kiefer. Sequential minimax search for a maximum. *Proceedings of the American mathematical society*, 4(3):502–506, 1953.

[14] Kenneth Lange. *Optimization*. Springer, 2004.

[15] Kenneth Lange. *Numerical analysis for statisticians*. Springer Science & Business Media, 2010.

[16] Kenneth Lange. The MM algorithm. `https://www.stat.berkeley.edu/~aldous/Colloq/lange-talk.pdf`, April 2007. [Online, accessed 18-September-2019].

[17] Randall J LeVeque. *Finite difference methods for ordinary and partial differential equations: steady-state and time-dependent problems*, volume 98. Siam, 2007.

[18] Steve McConnell. *Code complete*. Pearson Education, 2004.

[19] Geoffrey McLachlan and Thriyambakam Krishnan. *The EM algorithm and extensions*, volume 382. John Wiley & Sons, 2007.

[20] J Nocedal and SJ Wright. *Numerical Optimisation*. Springer verlag, 1999.

[21] Naoki Osada. *Acceleration methods for slowly convergent sequences and their applications*. PhD thesis, PhD thesis, Nagoya University, 1993.

[22] Yudi Pawitan. *In all likelihood: statistical modelling and inference using likelihood*. Oxford University Press, 2001.

[23] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2013. ISBN 3-900051-07-0.

[24] Walter Rudin et al. *Principles of mathematical analysis*, volume 3. McGraw-hill New York, 1964.

[25] Christopher G Small. A survey of multidimensional medians. *International Statistical Review/Revue Internationale de Statistique*, pages 263–277, 1990.

[26] Karline Soetaert, Thomas Petzoldt, and R. Woodrow Setzer. Solving differential equations in R: Package deSolve. *Journal of Statistical Software*, 33(9):1–25, 2010.

[27] Keller Vandebogart. Method of quadratic interpolation. `http://people.math.sc.edu/kellerlv/Quadratic_Interpolation.pdf`, September 2017. [Online; accessed 13-September-2019].

[28] Jet Wimp. *Sequence transformations and their applications*. Elsevier, 1981.

[29] Stephen Wright. Optimization for data analysis. In Michael W. Mahoney, John C. Duchi, and John C. Duchi, editors, *The Mathematics of Data*, chapter 2, pages 49–98. American Mathematical Society and IAS/Park City Mathematics Institute and Society for Industrial and Applied Mathematics, 2018.

[30] Tong Tong Wu, Kenneth Lange, et al. The MM alternative to EM. *Statistical Science*, 25(4):492–505, 2010.