Lock in Feedback in Sequential Experiments

Maurits Kaptein and Davide Iannuzzi

Abstract: We often encounter situations in which an experimenter wants to find, by sequential experimentation, \( x_{\text{max}} = \arg \max_x f(x) \), where \( f(x) \) is a (possibly unknown) function of a well controllable variable \( x \). Taking inspiration from physics and engineering, we have designed a new method to address this problem. In this paper, we first introduce the method in continuous time, and then present two algorithms for use in sequential experiments. Through a series of simulation studies, we show that the method is effective for finding maxima of unknown functions by experimentation, even when the maximum of the functions drifts or when the signal to noise ratio is low.

1. Introduction

We often encounter situations in which an experimenter wants to find, by sequential experimentation, \( \hat{x}_{\text{max}} = \arg \max_{\hat{x}} y = f(\hat{x}) \), where \( f(\hat{x}) \) is a (possibly unknown) function of a well controllable variable \( \hat{x} \). To solve this problem in physics and engineering applications, scientists have been routinely implemented techniques that rely on the idea of systematically changing the value of the controllable variable in time and following, via so-called lock-in amplifier techniques, how those changes affect the dependent variable \( y \). Applications of the problem of finding \( \arg \max_x f(x) \) are manifold and prominent in a much wider range of fields. For example, in economics, firms might be able to manipulate features of a product (\( x \)), such as the price, and observe the revenue (\( y \)). The exact functional relationship between product features and revenue might not be known but can sequentially be sampled (for examples see Kung et al., 2002; Jiang et al., 2011) (especially using modern interactive communication technologies, such as, for instance, e-commerce). The obvious aim of the firm is to select the product features that maximize the revenue. The problem also presents itself in the medical sciences when considering the dose of medication or composition of nutrition: often researchers seek for an optimal dosage according to some observable criterion, but the functional relationship between the dosage and the outcome measure is unknown (see, e.g., Sapareto and Dewey, 1984; Marschner, 2007). Despite the fact that the problem presents itself in many places, there is however no single agreed upon method to approach it. It is thus worth asking whether it be possible to adapt the lock-in a amplifier algorithm used in physics and engineering as a generic tool to find \( \hat{x}_{\text{max}} = \arg \max_{\hat{x}} y = f(\hat{x}) \).

*D.I. acknowledges the support of the European Research Council (grant agreement n. 615170) and of the Stichting voor Fundamenteel Onderzoek der Materie (FOM), which is financially supported by the Netherlands organization for scientific research (NWO).
The goal of this paper is to address this topic. We focus on the simple case where \( x \) is a scalar. In the remainder of this paper, we index sequential trials by \( t \in \{1, \ldots, T\} \) and our ultimate aim is to describe a novel method for manipulating \( x_t \) (in discrete time) to find, sequentially, the value of \( x \) that maximizes \( y \). We coin the method Lock-in Feedback (LiF).

The problem of finding \( \arg \max_x f(x) \) is treated in a number of branches in the experimental design and machine learning literature. The problem can be approached as an optimal design problem, in which the main aim is to design an experiment that efficiently provides us with information regarding \( f(x) \) (see, e.g., O’Brien and Funk, 2003; Myung and Pitt, 2009a). However, one could also frame the problem as a bandit problem (Berry and Fristedt, 1985), where the actions \( a \in A \) are the choice for the (continuous) value of \( x \), and the rewards at each interaction \( r_t \) are (some function of) \( y \) (Bubeck et al., 2011). In this paper we focus on the introduction and examination of our novel approach to this specific problem, and defer the embedding within the bandit and optimal design literature to the discussion Section.

As mentioned above, our method is based on an approach used in physics and engineering, where, if a physical variable \( y \) depends on the value of a well-controllable physical variable \( x \), the search for \( \arg \max_x f(x) \) can be solved via what is nowadays considered as standard electronics. This approach relies on the possibility of making the variable \( x \) oscillate at a fixed frequency and to look at the response of the dependent variable \( y \) at the very same frequency by means of a lock-in amplifier (Meade, 1983). The method is particularly suitable when \( y \) is immersed in a high noise level, where other more direct methods would fail. Furthermore, should the entire curve shift (or, in other words, if \( \arg \max_x f(x) \) changes in time, also known as concept drift), the circuit will automatically adjust to the new situation and quickly reveal the new maximum position. This approach is widely used in a very large number of applications, both in industry and research (Meade, 1983), and is the basis for the Lock-in Feedback (LiF) method we introduce in this paper.

In the following sections we first introduce LiF in more detail. We present an analysis of LiF in continuous time and demonstrate how an oscillating manipulation of the independent variable \( x \) can be used as a strategy to find \( \arg \max_x y = f(x) \). Subsequently we present two algorithms to use LiF in sequential experiments. We then, by simulation, compare the two algorithms, and examine the performance of LiF in several scenario’s of signal-to-noise ratio and in situations of concept drift (e.g., Gaber et al., 2005). Finally we discuss the opportunities that LiF may offer for future experiments and a number of open questions regarding this proposed experimental regime.

2. Finding the maximum of a curve with a lock-in algorithm: a short introduction

In this section we detail the basic principles behind LiF assuming continuous time in which \( x \) can be manipulated. Let’s assume that \( y \) is a continuous function
Let’s further assume that $x$ oscillates with time according to:

$$x(t) = x_0 + A \cos(\omega t)$$  \hspace{1cm} (1)

where $\omega$ is the angular frequency of the oscillation, $x_0$ its central value, and $A$ its amplitude. For relatively small values of $A$, Taylor expanding $f(x)$ around $x_0$ to the second order, one obtains:

$$y(x(t)) = f(x_0) + (x_0 - x_0 - A \cos(\omega t)) \left( \frac{\partial f}{\partial x} \Bigg|_{x=x_0} \right) + \frac{1}{2} (x_0 - x_0 - A \cos(\omega t))^2 \left( \frac{\partial^2 f}{\partial x^2} \Bigg|_{x=x_0} \right)$$  \hspace{1cm} (2)

which can be simplified to:

$$y(x(t)) = k - A \cos(\omega t) \left( \frac{\partial f}{\partial x} \Bigg|_{x=x_0} \right) + \frac{1}{4} A^2 \cos(2\omega t) \left( \frac{\partial^2 f}{\partial x^2} \Bigg|_{x=x_0} \right)$$  \hspace{1cm} (3)

where $k = f(x_0) + 1/4 \left( \partial^2 f/\partial x^2 \Bigg|_{x=x_0} \right)$. It is thus evident that, for small oscillations, $y$ becomes the sum of three terms: a constant term, a term oscillating at angular frequency $\omega$, and a term oscillating at angular frequency $2\omega$.

Suppose we ourselves can actively manipulate $x$ and measure $y$, and that $f$ is continuous and only has one maximum and no minimum. Further suppose that one is interested to find the value $\arg \max_x y = f(x)$ which we denote with $x_{max}$, and that our measurements of $y$ contain noise

$$y(t) = f(x(t)) + \epsilon$$  \hspace{1cm} (4)

where $\epsilon$ denotes the noise and $\epsilon \sim \pi()$ where $\pi$ is some probability density function and $\mathbb{E}[\epsilon|x] = 0$.

Following the scheme used in physical lock-in amplifiers (see, e.g., Scofield, 1994), we multiply the observed $y$ variable by $\cos(\omega t)$. Using eq. 3 and eq. 4, one obtains:

$$y_{\omega}(t) = \cos(\omega t) \left[ k - A \cos(\omega t) \left( \frac{\partial f}{\partial x} \Bigg|_{x=x_0} \right) \right. \left. + \frac{1}{4} A^2 \cos(2\omega t) \left( \frac{\partial^2 f}{\partial x^2} \Bigg|_{x=x_0} \right) \right. + \left. \epsilon \right]$$  \hspace{1cm} (5)

For simplicity of exposure we only consider these well-behaved functions in this paper.
where $y_\omega$ is the value of $y$ after it has been multiplied by $\cos(\omega t)$. Eq. 5 can be written more compactly as:

$$y_\omega = -\frac{A}{2} \left. \frac{\partial f}{\partial x} \right|_{x=x_0} + k_\omega \cos(\omega t) + k_{2\omega} \cos(2\omega t) + k_{3\omega} \cos(3\omega t) + \epsilon \cos(\omega t)$$

(6)

where

$$k_\omega = k + \frac{A^2}{2} \left( \left. \frac{\partial^2 f}{\partial x^2} \right|_{x=x_0} \right)$$

(7)

$$k_{2\omega} = -\frac{A}{2} \left( \left. \frac{\partial f}{\partial x^2} \right|_{x=x_0} \right)$$

(8)

$$k_{3\omega} = \frac{A^2}{8} \left( \left. \frac{\partial^2 f}{\partial x^2} \right|_{x=x_0} \right).$$

(9)

Integrating $y_\omega$ over a time $T = \frac{2\pi N}{\omega}$, where $N$ is a positive integer and $T$ denotes the time needed to integrate $N$ full oscillations, one obtains:

$$y_\omega^* = -\frac{TA}{2} \left. \frac{\partial f}{\partial x} \right|_{x=x_0} + \int_0^T \epsilon \cos(\omega t) \, dt$$

(10)

Depending on the noise level, one can tailor the integration time, $T$, in such a way to reduce the second addendum of the right hand of eq. 10 to negligible levels, effectively averaging out the noise in the measurements. Under those circumstances, $y_\omega^*$ provides a direct measurement of the value of the first derivative of $f$ at $x = x_0$.

The above method thus yields quantitative information regarding the first derivative of $f$ at $x = x_0$, providing, in this way, a logical update strategy of $x_0$: if $y_\omega^* < 0$, then $x_0$ is larger than the value of $x$ that maximizes $f$; likewise, if $y_\omega^* > 0$, $x_0$ is smaller than the value of $x$ that maximizes $f$. Thus, based on the oscillation observed in $y_\omega$ we are now able to move $x_0$ closer to $x = \arg\max_x f(x)$ using an update rule $x_0 := x_0 + \gamma y_\omega^*$ where $\gamma$ quantifies the learn rate of the procedure. Hence, we can setup a feedback loop that allows us to keep $x_0$ close to $x_{\text{max}}$, even if $f(x)$ changes over time.

Note that, multiplying $y$ by $\cos 2\omega t$ and using a similar approach as the one described above to extract the amplitude of the oscillation of $y$ at frequency $2\omega$, one would be able to measure the second derivative of the function $f$ at $x = x_0$. This property can be useful when, for instance, $f(x)$ is known to be an exact parabola to not only derive the direction of the step towards the maximum, but to work out the exact step size (see Appendix A).

3. Algorithm for LiF in experiments

In practical terms, measurements can never run in continuous mode. Therefore, we now present an algorithm for LiF in discrete time. To simplify notation,
we will index sequential measurements by \( y_t \) where \( t = 1, \ldots, t = T \) where \( T \) denotes the length—possibly infinite—of the experiment that is ran to find \( \text{arg max}_x f(x) \).

In discrete time we can use the same procedure as above in which we start with \( x_0 \), and for each sample oscillate around \( x_0 \) with a known frequency \( \omega \) and known amplitude \( A \):

\[
x_t = x_0 + A \cos \omega t
\]

which will result in measurements given by

\[
y_t = f(x_0 + A \cos \omega t) + \epsilon_t
\]

On the basis of the arguments reported above, we can now implement a feedback loop that iteratively adjusts the value of \( x_0 \) until \( x \) reaches \( x_{\text{max}} \). After that, if the function \( f \) changes, the loop can follow the value of \( x \) to the new maximizing position and thus stay "locked". The procedure is similar to that given in Equation 6 and 10, where we first multiply the outcome \( y_t \) by \( \cos(\omega t) \) and subsequently integrate out the noise term (summing in the discrete case). In the following sections we present two possible implementations for LiF in discrete time for use in sequential experiments.

3.1. LiF-I: Batch updates of \( x_0 \)

Our first implementation of LiF (denoted LiF-I) is presented in Algorithm 1. In this implementation we summate observations \( y_t \), which we multiply by \( \cos(\omega t) \), for a batch period of length \( T \), after which we update \( x_0 \). Variable \( y^*_\omega \) contains a running sum that is used for the integration.

\begin{algorithm}
\caption{Implementation of LiF-I for single variable maximization in data stream using a batch approach.}
\begin{algorithmic}
\REQUIRE \( x_0, A, T, \gamma, y^\Sigma_\omega = 0 \)
\STATE \( \omega = \frac{2\pi}{T} \)
\FOR {\( t = 1, \ldots, T \)}
\STATE \( x_t = x_0 + A \cos \omega t \)
\STATE \( y_t = f(x_0 + A \cos \omega t) + \epsilon_t \)
\STATE \( y^\Sigma_\omega = y^\Sigma_\omega + y_t \cos \omega t \)
\IF {\( (t \mod T == 0) \)}
\STATE \( y^*_\omega = \frac{y^\Sigma_\omega}{T} \)
\STATE \( x_0 = x_0 + \gamma y^*_\omega \)
\STATE \( y^\Sigma_\omega = 0 \)
\ENDIF
\ENDFOR
\end{algorithmic}
\end{algorithm}

The tuning parameters for LiF-I, which should be set by the experimenter, are \( x_0, A, T, \gamma \). Here below we describe some general criteria the choice may be based on:
• It is advised to set \( x_0 \) as close as possible to \( x_{\text{max}} \). The choice can only be based on the available information on \( f \). The more accurate the information, the closer the initial \( x_0 \) to \( x_{\text{max}} \), the faster the convergence of the loop to \( x_{\text{max}} \).

• The amplitude \( A \) affects the costs of the search procedure, because a large \( A \) implies querying a large range of \( x \) values with (possibly) low resulting \( y \) values. However, \( A \) also influence the learning speed: a very small \( A \) leads to small updates steps, while a large value of \( A \) might lead to a value of \( \gamma y_{\omega} \) that “overshoots” \( x_{\text{max}} \).

• The integration time \( T \) affects the variability of the update of \( x_0 \), with larger integration times leading to a smoother update but slower convergence.

• The learn-rate \( \gamma < 1 \) determines the step size at each update of \( x_0 \). This can be interpreted, and tuned, akin learn-rates in, for instance, stochastic gradient descent methods (Poggio et al., 2011).

3.2. LiF-II: Continuous updates of \( x_0 \)

For some applications the batch updates of \( x_0 \) – as implied by the continuous time analysis and defined in Algorithm 1 – might not be feasible. Algorithm 2 presents a modified version of LiF (denoted LiF-II) in which \( x_0 \) is updated every observation. LiF-II starts by filling up a buffer of length \( T \) which we denote by the vector \( \vec{y}_{\omega} = \{NA_1, \ldots, NA_T\} \), after which each observation leads to an update of \( x_0 \). In the algorithm description the values \( y_{t-T}, \ldots, y_t \) are stored in the vector \( \vec{y}_{\omega} \). By defining the learn rate as \( \frac{\gamma_T}{T} \) the tuning parameters in LiF-II are the same as those discussed for LiF-I.

\textbf{Algorithm 2} Implementation of LiF-II for single variable maximization in data stream using a batch.

\textbf{Require:} \( x_0, A, T, \gamma, \vec{y}_{\omega} = \{NA_1, \ldots, NA_T\} \)

\begin{algorithmic}
\State \( \omega = \frac{2\pi}{T} \)
\For {\( t = 1, \ldots, T \)}
\State \( x_t = x_0 + A \cos \omega t \)
\State \( y_t = f(x_0 + A \cos \omega t) + \epsilon_t \)
\State \( \vec{y}_{\omega} = \text{push}(\vec{y}_{\omega}, y_t \cos \omega t) \)
\If {\( t > T \)}
\State \( y_\omega = (\sum \vec{y}_{\omega})/T \)
\State \( x_0 = x_0 + \frac{\gamma_T}{T} y_\omega \)
\EndIf
\EndFor
\end{algorithmic}

4. Simulation study 1: Comparison of Batched and streaming LiF and examination of tuning parameters

In this section we study, by simulation, the differences between LiF-I and LiF-II, and the effects of the tuning parameters \( A, T, \) and \( \gamma \) in a situation in which \( y = f(x) \) is measured without noise.
Figure 1. Examination of the effect of tuning parameters $A$ and $T$ for $\gamma = .1$. Displayed are the results for LiF-I (black solid line) and LiF-II (gray dotted line).

Figure 1 presents the performance of both LiF-I and LiF-II for data generated using

$$f(x) = -2(x - 5)^2 + \epsilon$$  \hspace{1cm} (13)

where $\epsilon \sim \mathcal{N}(0, 0)$. The figure displays the performance of LiF for $T = 10000$ using the following tuning parameter settings

- $x_0 = -5$.
- $T \in \{10, 100, 1000\}$
- $A \in \{.1, 1, 2, 10\}$
- $\gamma = .1$

The rows of Figure 1 (top to bottom) present decreasing values of $\gamma$, while the columns (left to right) present increasing values of $T$. We fix $A = 1$. Each panel presents the value of $x_0$ during the data stream as selected using LiF-I (black solid line) and LiF-II (gray dotted line). It is clear that LiF can “overshoot” the maximum for values of $\gamma$ that are too high (top two rows). This happens for both
LiF-I and LiF-II, although LiF-I seems more robust. For small values of $\gamma$ the performance of the algorithms is very similar, and increases in the integration window $T$ merely smooth the updating procedure.

In Figure 2 the results are plotted for the same setup, but this time we vary $A$, while $\gamma = 1$. Here it is clear that for large values of $A$ LiF-I has a tendency to become unstable (see top rows), while the streaming LiF-II is much more robust for erroneous selection of $A$. Very small choices for the amplitude $A$ lead to very slow updates of $x_0$ in both cases. Again, increased in $T$ merely smooth the process. The simulations give an impression of the importance of the tuning parameters $x_0$, $A$, $T$, $\gamma$ and their relationships. In the remainder of this paper we will focus on the evaluation – through simulation – of the performance of LiF-II in cases of noise and concept drift.

5. Simulation study 2: Effects of noise

To examine the impact of (measurement) noise on the performance of LiF-II we repeat the simulations as described in Section 4 using the data generating model
described by Equation 14 with $\epsilon \sim \mathcal{N}(0, \sigma^2)$ and $\sigma^2 \in \{10, 100, 1000, 10000\}$. We choose tuning parameters: $x_0 = -5$, $A = 1$, $T = 100$, $\gamma = .1$. Contrary to the simulations presented in Section 4 we now repeat the procedure $m = 100$ times: Figure 3 presents the average $x_0$ over the 100 simulation runs as well as the 95% confidence bounds. From Figure 3 it is clear that LiF-II performs very well also in the situation in which the noise levels are high.

6. Simulation study 3: Performance of LiF in cases of concept drift

One of the advantages of Lock in Feedback as opposed to other methods of finding $x_{max}$ is the fact that LiF can also be used to find a maximum of a function in cases of concept drift (Gaber et al., 2005): even when $f(x)$ changes over time, LiF provides a method to keep the value of the treatment $x$ close to $x_{max}$.

To illustrate this latter advantage of LiF we setup a simulation using the following data generating model:

$$f(x,t) = -2((x - .0025t) - 5)^2 + \epsilon$$

where the $(x - .0025t)$ ensures that during the stream running from $t = 0$ to $t = 10^4 = T$ the value of $x_{max}$ moves from 5 to 30. We choose $x_0 = -20$ (note the different starting position compared to the previous simulations), $A = 1$, $T = 100$, $\gamma = .1$ and $\sigma^2 = 10$. We investigate the performance of LiF-II in this case of concept drift.

Figure 4 presents in the top panel $y = f(x,t)$ for distinct values of $t \in \{0, 1000, \ldots, 10000\}$ in different shades of grey. The concept drift is illustrated by the different locations of the parabola. Superimposed in blue is the value of $x_0$ as selected by LiF-II. In the bottom panel the value of $x_0$ as a function of the length of the stream is presented. It is clear that LiF-II quickly finds $x_{max}$ and follows the maximum as it moves during the stream.
Figure 4. Illustration of LiF in the case of concept drift. As the true maximum shifts (top panel) LiF is able to follow the maximum and keep $x_0$ close to $x_{\text{max}}$. 
7. Discussion and Future work

In this paper we presented Lock in Feedback as a method to find $\arg\max_x f(x)$ through sequential experiments. The method is appealing since it a) does not require the functional form of $f(x)$ to be known to derive its maximum, b) performs well in situations in which measurements are obtained with large noise, and c) allows following the maximum of a function even if that function changes over time. We have presented the basic mathematical arguments behind LiF, demonstrating how known (or imposed) oscillations in $x$ can be used to determine the derivative(s) of $f(x)$ which can subsequently be used to find $\arg\max_x f(x)$. Next, we detailed two possible implementations of LiF and examined their performance for a variety of tuning parameter settings. We then showed that a streaming version of LiF is robust both to noise as well as concept drift.

The current expose of LiF is however rudimentary. Relationships and comparisons to existing methods of optimization in sequential experiments have not yet been carried out and comprise obvious future work. Furthermore, the ability to use LiF for problems of higher dimension, e.g., where $y = f(\vec{x})$ is a function of multiple variables, has not been explored even though this extension relatively is easily made. In the remained of this discussion section we try to address these concern in more detail and give suggestions for future examinations of LiF.

7.1. LiF for higher dimensional problems.

In this paper we have demonstrated the use of LiF only in cases where $x$ is scalar. However, when $x$ is a vector a very similar approach can be used to find the maximum of the function $f(\vec{x})$ in more than one dimension. In the two dimensional case LiF can be extended by oscillating both elements of $x$ at different frequencies:

\[
x_{1,t} = x_{1,0} + A_1 \cos \omega_1 t \\
x_{2,t} = x_{2,0} + A_2 \cos \omega_2 t
\]

After oscillating both elements of $x$ we observe $y_t = f(x_{1,t}, x_{2,t})$ and we can obtain information regarding the gradient by separately computing:

\[
y_{1,\omega} = y_t \cos \omega_1 t \\
y_{2,\omega} = y_t \cos \omega_2 t
\]

This simple extension allows for the use of LiF in higher dimensions. However, besides the fact that $\omega_1$ and $\omega_2$ should not be multiples of each other, the effects of the tuning parameters and the performance of this higher dimensional version of LiF need to be further examined.
7.2. LiF and optimal design

Often, the problem of finding \( \arg\max_x f(x) \) is solved by performing an experiment to be able to estimate \( f(x) \), after which the solution can be derived analytically. There is a large literature on designing optimal experiment to to estimate the parameters of \( f(x) \) when its functional form is known (Antille and Weinberg, 2000; Myung and Pitt, 2009b). Future work should examine the efficiency of LiF compared to methods in which one first carries out an (optimal) experiment to estimate \( f(x) \) before deriving \( \arg\max_x f(x) \). Also, the robustness to model misspecification should be examined: one advantage of LiF over methods in which \( f(x) \) is examined through experimentation is that \( f(x) \) need not be known anywhere in the process. The practical benefit of the ability of LiF to uncover \( x_{\text{max}} \) without prior knowledge regarding \( f(x) \) in specific situations however needs more scrutiny.

7.3. LiF in continuous bandit problems

The problem of finding \( \arg\max_x f(x) \) can also be cast as a bandit problem. In such a formalization the selected values of \( x \) present the actions chosen by the experimenter after which the rewards \( y \) are observed. Here, one is interested in finding a policy – a method to select values of \( x \) given the previous observations – which maximizes \( R = \sum y_t \).\(^2\) In such a setting one would not only be concerned about finding \( x_{\text{max}} \), but one would like to be efficient in the search. Efficiency here can be quantified in terms of regret: the performance of the search procedure can be compared to the \( R^* \) observed when \( x_{\text{max}} \) would be known.

LiF might be an efficient method of solving such a (continuous) bandit problem. However, for LiF to be an asymptotically optimal solution to the continuous valued bandit problem, the learning costs, which relate directly to the amplitude of the induced oscillation, should be decreased over time: \( A \to 0 \). However, LiF runs the risk of getting stuck in a local maximum thus suffering linear regret. Thus, the tuning parameter \( A \) relates directly to the exploration-exploitation behavior of LiF.

7.4. Conclusion

In this paper we introduced LiF as a method for finding \( \arg\max_x f(x) \). LiF is appealing since it a) does not require knowledge of \( f(x) \), b) performs well in situations with large noise, and c) works in cases of concept drift. We have provided two algorithms to implement LiF in the simple scalar case. However, future work should examine higher dimensional implementations of LiF, and relations to other known approaches to find \( \arg\max_x f(x) \).

\(^2\)For simplicity we are not discussing discounting or rewards that are a function of the observable \( y \).
References

Antille, G. and Weinberg, A. (2000). *A Study of D-optimal Designs Efficiency for Polynomial Regression*. Université de Genève, Faculté des sciences économiques et sociales, Département d’économétrie.

Berry, D. A. and Fristedt, B. (1985). *Bandit Problems: Sequential Allocation of Experiments*. Springer.

Bubeck, S., Munos, R., and Stoltz, G. (2011). Pure exploration in finitely-armed and continuous-armed bandits. *Theoretical Computer Science*, 412(19):1832–1852.

Gaber, M. M., Zaslavsky, A., and Krishnaswamy, S. (2005). Mining data streams. *ACM SIGMOD Record*, 34(2):18.

Jiang, Y., Shang, J., Kemerer, C. F., and Liu, Y. (2011). Optimizing E-tailer Profits and Customer Savings: Pricing Multistage Customized Online Bundles. *Marketing Science*, 30(4):737–752.

Kung, M., Monroe, K. B., and Cox, J. L. (2002). Pricing on the Internet. *Journal of Product & Brand Management*, 11(2):274–288.

Marschner, I. C. (2007). Optimal design of clinical trials comparing several treatments with a control. *Pharmaceutical statistics*, 6(1):23–33.

Meade, M. L. (1983). *Lock-in amplifiers: principles and applications*, volume 1. Mike Meade.

Myung, J. and Pitt, M. (2009a). Bayesian adaptive optimal design of psychology experiments. *Proceedings of the 2nd International Workshop in Sequential Methodologies (IWSM2009)*, pages 1–6.

Myung, J. I. and Pitt, M. A. (2009b). Optimal experimental design for model discrimination. *Psychological review*, 116(3):499.

O’Brien, T. E. and Funk, G. M. (2003). A gentle introduction to optimal design for regression models. *The American Statistician*, 57(4):265–267.

Poggio, T., Voinea, S., and Rosasco, L. (2011). Online Learning, Stability, and Stochastic Gradient Descent. *Artificial Intelligence*, 8(11):11.

Sapareto, S. A. and Dewey, W. C. (1984). Thermal dose determination in cancer therapy. *International Journal of Radiation Oncology*Biology*Physics*, 10(6):787–800.

Scofield, J. H. (1994). Frequency-domain description of a lock-in amplifier. *American Journal of Physics*, 62(2):129.

Appendix A: Algorithm for finding the exact maximum of a parabola using the second order approximation.

Let’s know suppose that the curve $y = f(x)$ is a parabola:

$$y = -\alpha(x - x_0)^2 + \gamma$$

Clearly, $f(x)$ has a maximum for $x = x_0$. Furthermore, the second derivative is always equal to $-2\alpha$, regardless the value of $x$. Interestingly, the value of $\alpha$ can
be easily extracted from the data accumulated during the lock-in procedure. For this purpose, $y(t)$ has to be multiplied by $\cos(2\omega t)$. Following the steps illustrated in eq. 5, eq. 6, and eq. 10, one obtains:

$$y_{2\omega} = \frac{TA^2}{8} \left( \frac{\partial^2 f}{\partial x^2} \bigg|_{x=x_0} \right)$$

$$+ \int_0^T \epsilon \cos(2\omega t)$$

which allows us to calculate $\alpha$ as:

$$\alpha = \frac{4y_{2\omega}}{TA^2}$$