James Akl May 2020

Noise Extraction in Disturbance Observer-Based Control

A Computational Inverse Problems Approach

Introduction

In certain applications, there is great interest in estimating noise as it manifests itself in the output. One such application is 'Disturbance Observer-Based Control' (DOBC). In DOBC, the eventual goal is disturbance rejection (*i.e.*, removing noise or perturbation called disturbance from the control loop). This is especially useful when higher-order dynamics are intractable or too expensive to compute [1]. The unmodeled dynamics and modeling errors are lumped as a second, unknown input (the disturbance), and a device called 'disturbance observer' estimates the disturbance. When the estimate is good enough, it can be subtracted from the input to reduce the effect of the true disturbance. There is great interest in applying DOBC as a cheap and reliable alternative to expensive higher-order formulations, especially in manufacturing, robotics, and aerospace [2].

Inverse Problem Structure

The system G needs to be controlled, meaning its actual output $\mathbf{y}$ should match some arbitrary reference target (the desired output) $\mathbf{y}_{\mathbf{ref}}$. G can be represented in different ways *e.g.*, state-space, transfer function, integral equation, matrix, or a general mapping. The controller generates an adequate control input $\mathbf{u}$ which maps to $\mathbf{y}$ via G *i.e.*, $\mathrm{G}\mathbf{u} = \mathbf{y}_{\mathbf{u}}$. In other words, the controller executes $\mathbf{y}_{\text{ref}} \mapsto \mathbf{u}$ and provides G with an input $\mathbf{u}$. The controller is assumed ideal and is abstracted away in this problem *i.e.*, $\mathbf{u}$ is given directly without needing $\mathbf{y}_{\text{ref}}$. Unknown inputs $\mathbf{d}$ (noise, perturbation, disturbance) get added to the input $\mathbf{u}$ and the sum is mapped by G as in (f_1):

$$\mathbf{u}(t) + \mathbf{d}(t) \quad \mapsto_G \quad \mathbf{y}_{\mathbf{u+d}}(t) \tag{f_1}$$

A disturbance estimate $\hat{\mathbf{d}}(t)$ is pre-subtracted from the control input $\mathbf{u}(t)$ giving (f_2):

$$\mathbf{u}(t) + \mathbf{d}(t) - \hat{\mathbf{d}}(t) \quad \mapsto_G \quad \hat{\mathbf{y}}(t) \tag{f_2}$$

$\hat{\mathbf{y}}(t)$ is the estimated output. Clearly, as $\hat{\mathbf{d}} \to \mathbf{d}$ then $\hat{\mathbf{y}} \to \mathbf{y}_{\mathbf{u}}$. An important distinction is that both the control inputs and noisy outputs $\mathbf{u}(t)$ and $\mathbf{y}_{\mathbf{u+d}}(t)$ are known. The goal is to estimate the *unknown input i.e.*, $\mathbf{d}(t)$, using $\hat{\mathbf{d}}(t)$, given $\mathbf{u}(t)$ and $\mathbf{y}_{\mathbf{u+d}}(t)$. Disturbance is not necessarily random noise and can be unmodeled dynamics with a sinusoidal profile, or in the simplest case, a constant offset. The continuous signals can be discretized *i.e.*, $\mathbf{u}(t) \in U$ is replaced with $\mathbf{u} \in \mathbb{R}^n$ by sampling n points.

Additional rewriting is performed to restructure into a common Inverse Problems formulation. The problem essentially is (f_3).

$$\hat{\mathbf{d}} = \underset{\mathbf{d}}{\operatorname{argmin}} \|\mathbf{y}_{\mathbf{u}+\mathbf{d}} - \mathrm{G}(\mathbf{u}+\mathbf{d})\| \tag{f$_3$}$$

For clarification, the notation is equivalent to:

$$\mathbf{x}_{\text{true}} \equiv \mathbf{u}+\mathbf{d} \qquad \mathbf{x} \equiv \mathbf{u}+\hat{\mathbf{d}} \qquad \mathrm{G} \equiv \mathrm{G} \qquad \mathbf{y}_{\text{true}} \equiv \mathrm{G}(\mathbf{u}+\mathbf{d}) \qquad \mathbf{y} \equiv \mathrm{G}\big(\mathbf{u}+\hat{\mathbf{d}}\big) = \mathbf{y}_{\mathbf{u}+\mathbf{d}}$$

Stating and solving $\mathbf{y} = \mathrm{G}(\mathbf{x})$ yields (f_5) where $\boldsymbol{\varepsilon}$ is the "output noise", *i.e.*, the deviation from the true output. In this case it refers to the deviation produced by an imperfect estimate $\hat{\mathbf{d}}$.

$$\mathbf{y}_{\mathbf{u}+\mathbf{d}} = \mathrm{G}\big(\mathbf{u}+\hat{\mathbf{d}}\big) = \mathbf{y}_{\text{true}} + \boldsymbol{\varepsilon} = \mathrm{G}(\mathbf{u}+\mathbf{d}) + \boldsymbol{\varepsilon} \tag{f$_5$}$$

From this, obtain the residual,

$$\mathbf{r} = \mathbf{y} - \mathrm{G}(\mathbf{x}) \equiv \mathbf{y}_{\mathbf{u}+\mathbf{d}} - \mathrm{G}\big(\mathbf{u}+\hat{\mathbf{d}}\big) \tag{f$_6$}$$

Thus, obtain the naive solution using the least-squares formulation,

$$\hat{\mathbf{d}}_{\text{LS}} = \underset{\hat{\mathbf{d}}}{\operatorname{argmin}} \big\|\mathbf{y}_{\mathbf{u}+\mathbf{d}} - \mathrm{G}\big(\mathbf{u}+\hat{\mathbf{d}}\big)\big\|_2^2 \tag{f$_7$}$$

The LS formulation in (f_6) is equivalent, in form, to,

$$\mathbf{x}_{\text{LS}} = \underset{\mathbf{x}}{\operatorname{argmin}} \|\mathbf{y} - \mathrm{G}(\mathbf{x})\|_2^2 \tag{f$_8$}$$

Usually, $\mathbf{x}$ can be directly obtained, whereas here $\hat{\mathbf{d}}$ is obtained via the expression $\mathbf{u}+\hat{\mathbf{d}}$.

Literature Review

All the literature used was devoted to better understanding the problem and setup of Disturbance Observer–Based Control. In effect, this problem has never been formulated in the structure treated in this report. More so, DOBC is usually studied in a purely continuous-signal approach and is only converted to discrete control in industrial applications. From [1], the contextual information and utility of DOBC can be found, and in [2] detailed formulations and examples illustrate the effectiveness of DOBC to compensate unknown inputs, whether deterministic, stochastic, linear, or nonlinear. A very brief and simple formulation of DOB controllers, as well as a broad exposition of major state and disturbance estimators and observers as used in control can be found in [3]. Inverse problem solution techniques are becoming more popular in control, although it is entirely absent from the realm of disturbance observers; a gap explored in this project.

Materials & Methods

Forward Operator

In most practical control applications, G is either linear or linearized to exploit the wealth of simple yet powerful tools within Linear Control Theory. G will be modeled as a Fredholm equation of the first kind (f_9), to be discretized. The inverse problem thus starts as **continuous linear** but ends up as **discrete linear**.

$$y(s) = \int_{t_1}^{t_n} g(s,t)\, u(t)\ \mathrm{d}t \tag{f_9}$$

Discretize in s and t with m and n steps respectively to obtain $\mathrm{G} \in \mathbb{R}^{m\times n}$, in which case, $\{\mathbf{u}, \mathbf{d}, \hat{\mathbf{d}}\} \subset \mathbb{R}^n$ and $\{\mathbf{y}_{\mathbf{u+d}}, \mathbf{r}, \hat{\mathbf{y}}\} \subset \mathbb{R}^m$. Finally, the discrete linear inverse problem is $\mathbf{y}_{\mathbf{u+d}} = \mathrm{G}(\mathbf{u} + \mathbf{d})$, where the eventual goal is to estimate $\mathbf{d}$ with $\hat{\mathbf{d}}$ given $\mathbf{u}$, $\mathbf{y}_{\mathbf{u+d}}$, and G.

The discretization and system generation are explained in the comments of `generate.m`.

Data

The problem will have the following components: $u(t) = \sin^2(\pi t)$ and $g(s,t) = H(s-t)$ where H is the Heaviside step function. Various disturbance types can be tested, most notably $d(t) \in \{\sin t\,, 1, \mathfrak{n} \sim \mathcal{N}(\mu, \sigma)\}$.

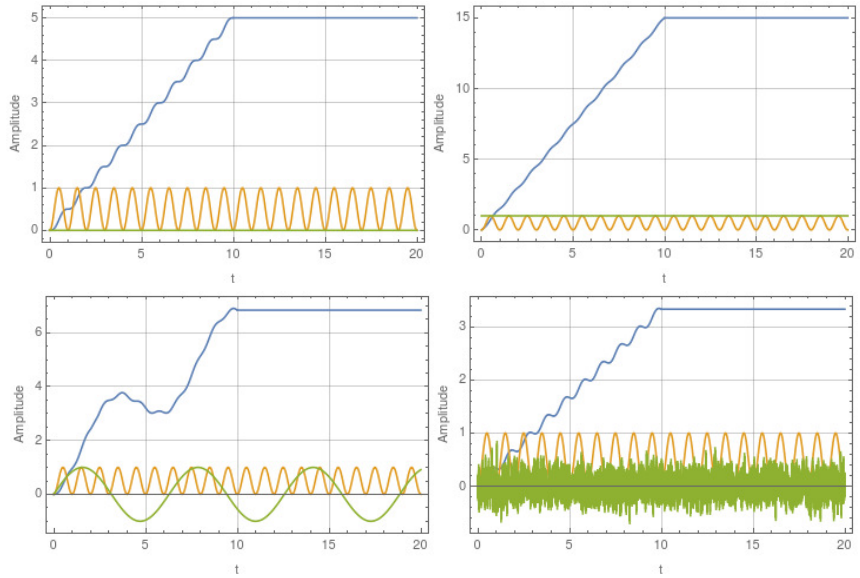
This setup simulates oscillations that accumulate to a threshold maximum after which the signal is clipped to a constant, similarly to what happens with a speaker and its membrane that cap at a maximum amplitude.

With $\mathbf{y}_{\mathbf{u+d}}$ and G given to us, estimate $\mathbf{u} + \hat{\mathbf{d}}$ using a subset of the methods listed below; and given $\mathbf{u}$ obtain $\hat{\mathbf{d}}$ and thus the estimated output $\hat{\mathbf{y}}$. Compare these to $\mathbf{d}$ and $\mathbf{y}_{\text{true}}$ and discriminate between the method outcomes.

In the plots below, the (orange) input, as noted is $u(t) = \sin^2(\pi t)$ and $g(s,t) = H(s-t)$. The (green) disturbance $d(t)$ is added and the (blue) output response $y(s)$ is obtained.

The disturbance type is laid out below in the same order as in the plot array (next page).

$$\begin{array}{ll} d(t) = 0 & d(t) = 1 \\ d(t) = \sin t & d(t) \sim \mathcal{N}(\mu = 0, \sigma = 0.2) \end{array}$$



The m-function `generate` has the header `function [G,y,u,d] = generate(N,q)` and generates all required quantities. Here, `N` is the size of the problem (note: must be even), specifically the size of the output-space. The output space is smaller (adding certain inconveniences to the problem). Also, `q` is the disturbance type selector, which chooses one of the four disturbance types above.

Since there are four disturbance types, and three solution methods, then there are 12 unique simulation cases. There are around four plots per case, which leaves us with an approximate total of 48 plots. Instead of displaying them in all, an illustrative subset is included.

If the remaining cases are of interest, it is trivial to change the simulation configuration. In the MATLAB `main` m-scripts (three total, one for each method), modify the disturbance type selector `q` as detailed in the m-script comments. For example. setting `q = 3` runs the simulation for Gaussian disturbance.

Methods

The inverse problems will be solved using three distinct methods.

- **Tikhonov regularization** via SVD, with the parameter $\alpha \in [0.001, 10]$.
- **Conjugate gradient least-squares**, with the parameter $k \in [1, n]$.
- **Stochastic gradient descent**, with the parameter k, in our problem $k \leq 20{,}000$.

Results

In the provided simulations, the problem dimensions are as follows:

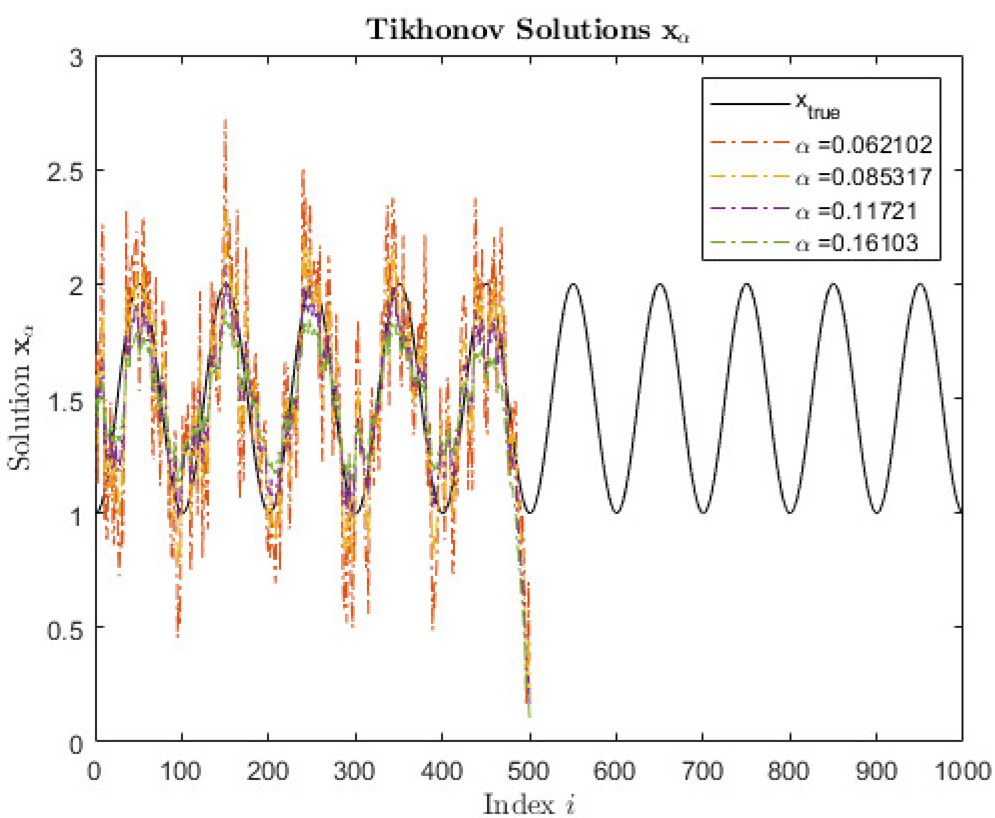
$$\mathrm{G} \in \mathbb{R}^{1000\times 500}, \qquad \{\mathbf{u}, \mathbf{d}\} \subset \mathbb{R}^{1000}, \qquad \{\mathbf{y}, \hat{\mathbf{y}}\} \subset \mathbb{R}^{500}$$

Note that for the sake of plotting, the outputs $\mathbf{y}$ are extended to $\mathbb{R}^{1000}$ for plotting purposes (to show the effect of the convolution, see derivation above).

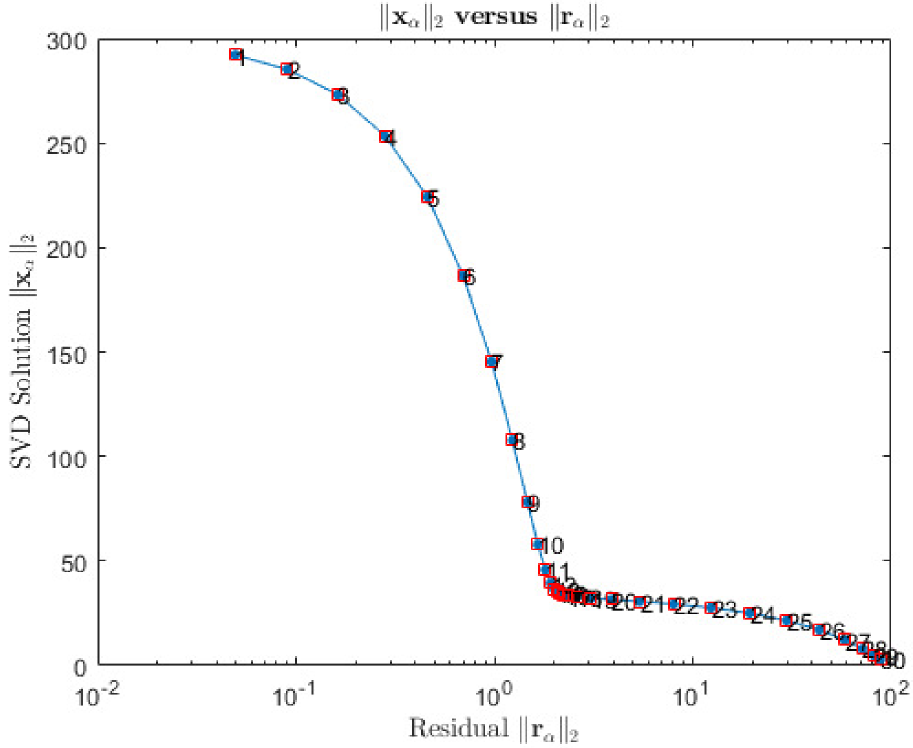
Note that, not all results for every single case are displayed, since it would make the report excessively long, and would not add additional insight. However, any desired case can be obtained by running the `main` m-scripts with the adequate disturbance type selector `q`.

Tikhonov Regularization

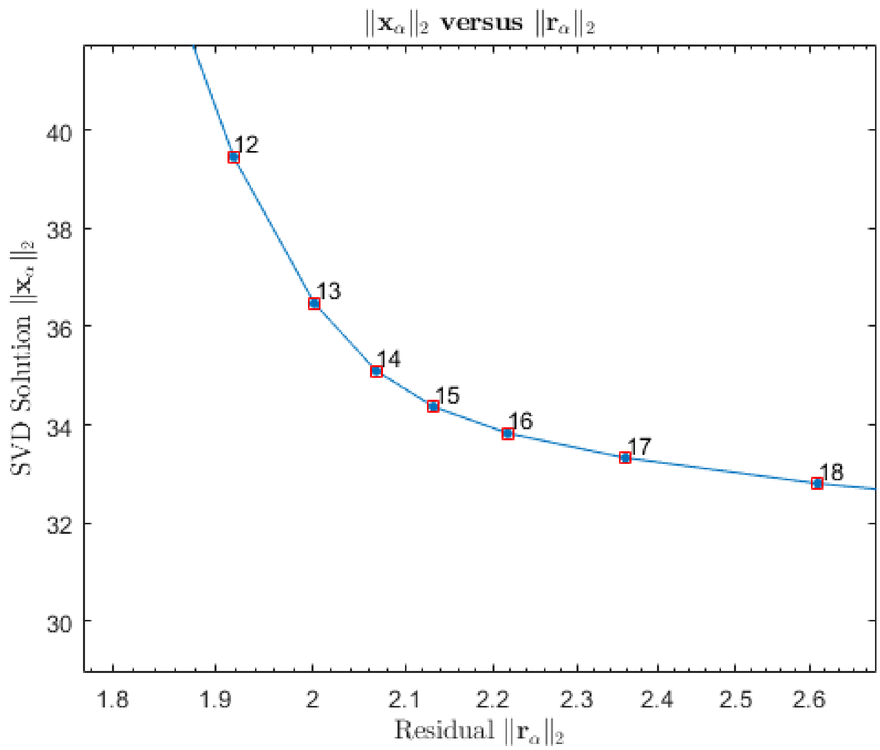
The following results are displayed for the 'best' Tikhonov solutions.



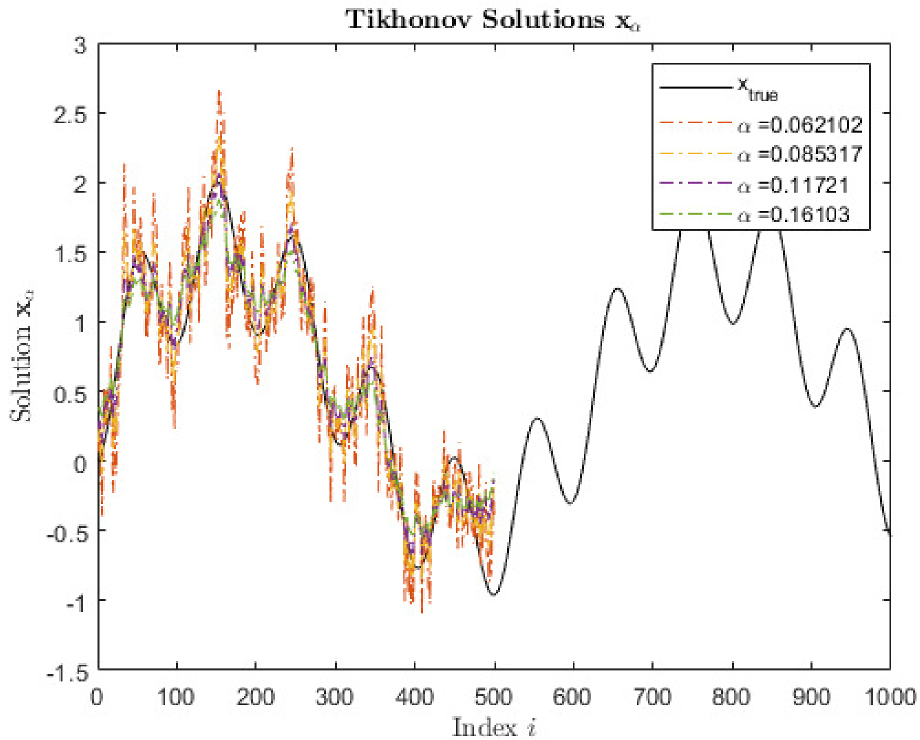
Sinusoidal Disturbance (`q = 1`)



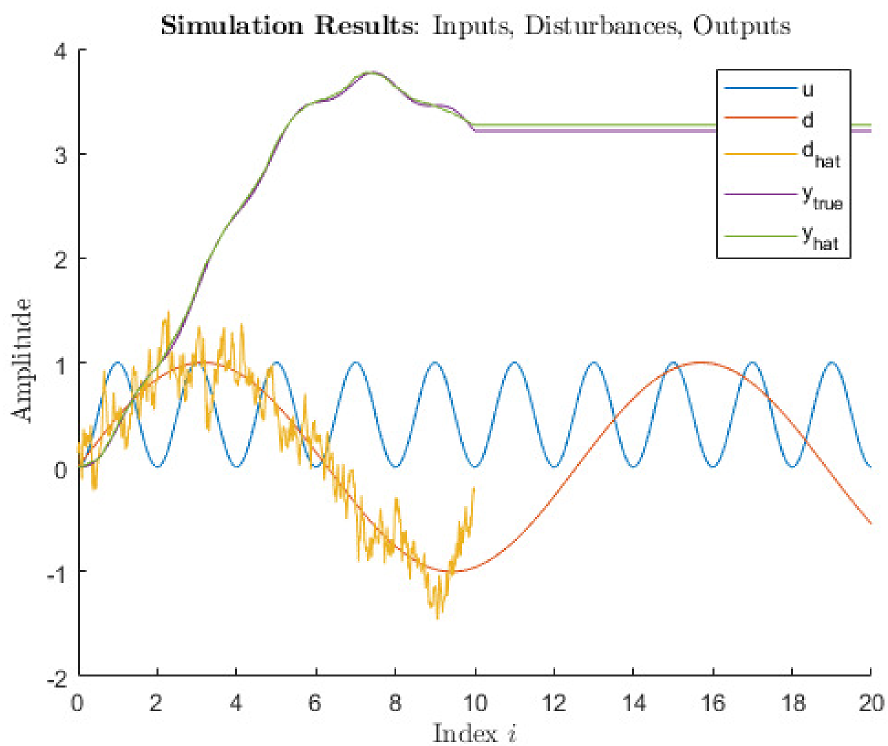
L–Curve (q = 1)



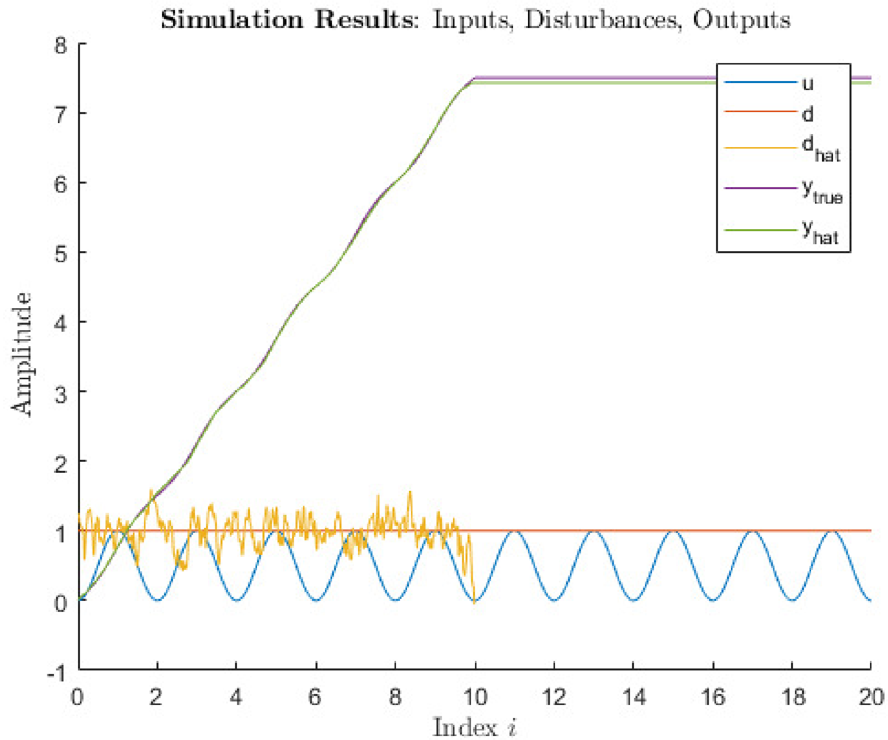
L–Curve (q = 1)



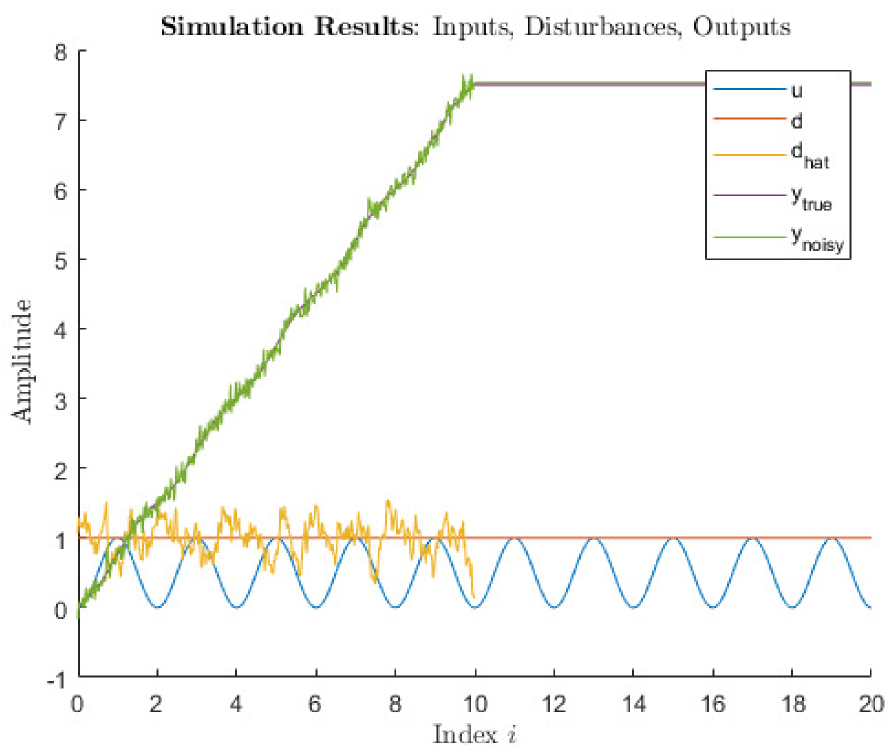
Sinusoidal Disturbance (`q = 2`)



Sinusoidal Disturbance (`q = 2`)



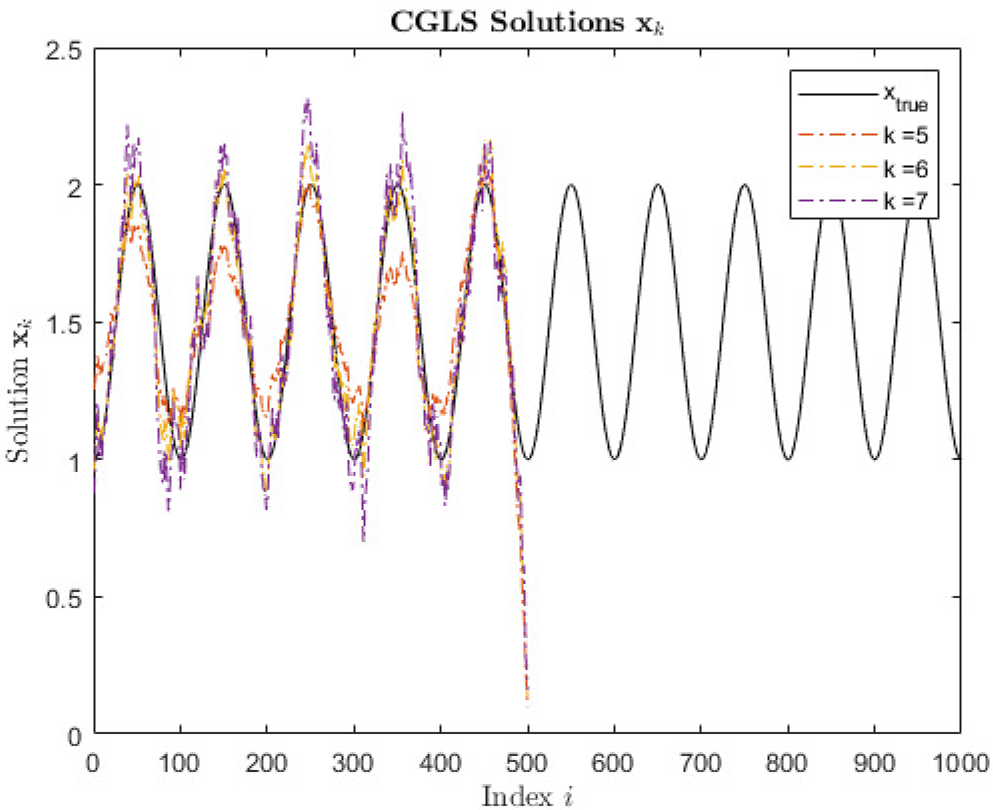
Constant Disturbance (`q = 1`)



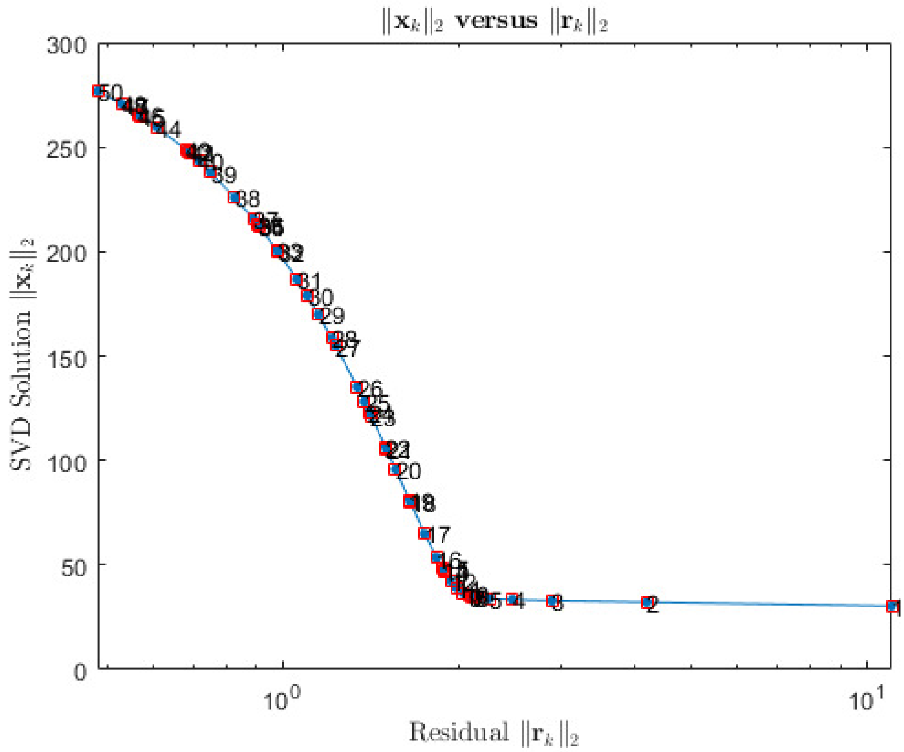
Constant Disturbance (`q = 1`)

Conjugate Gradient Least–Squares

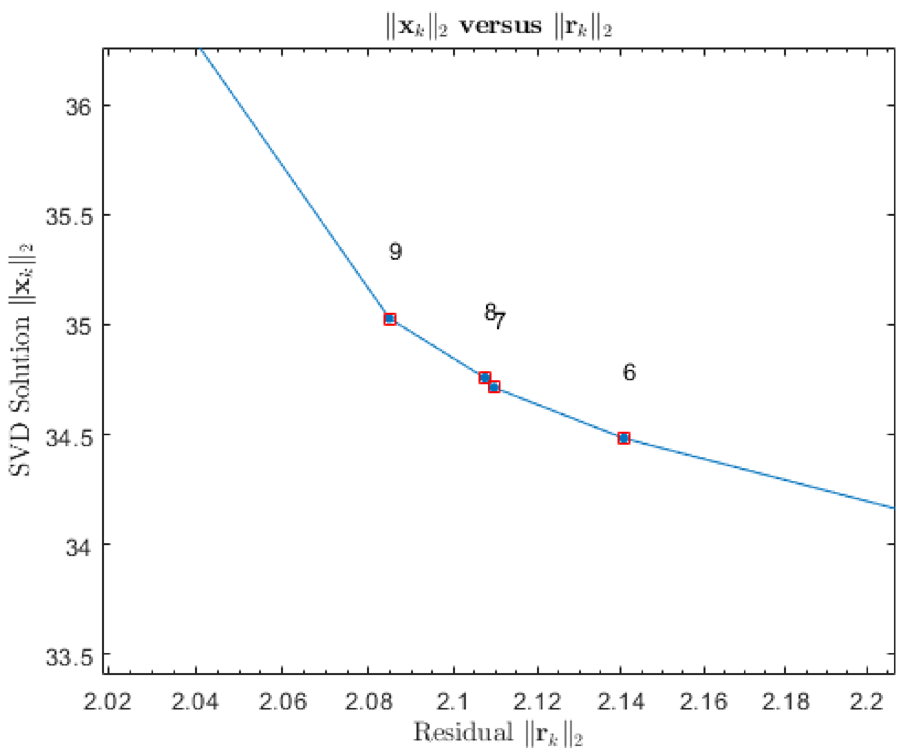
The following results are displayed for the 'best' CGLS solutions.



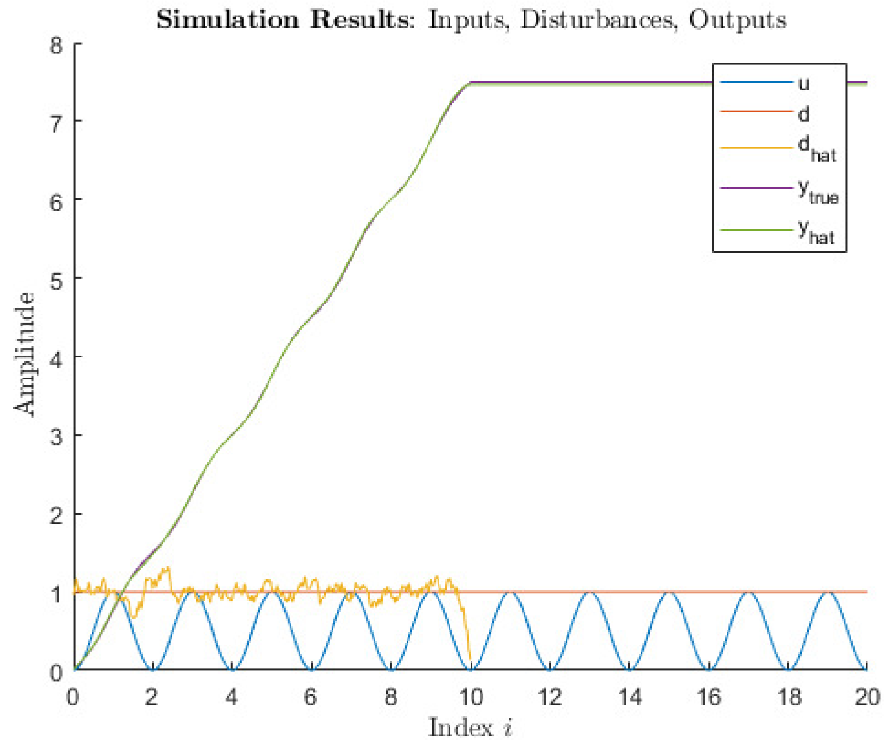
Constant Disturbance (`q = 1`)



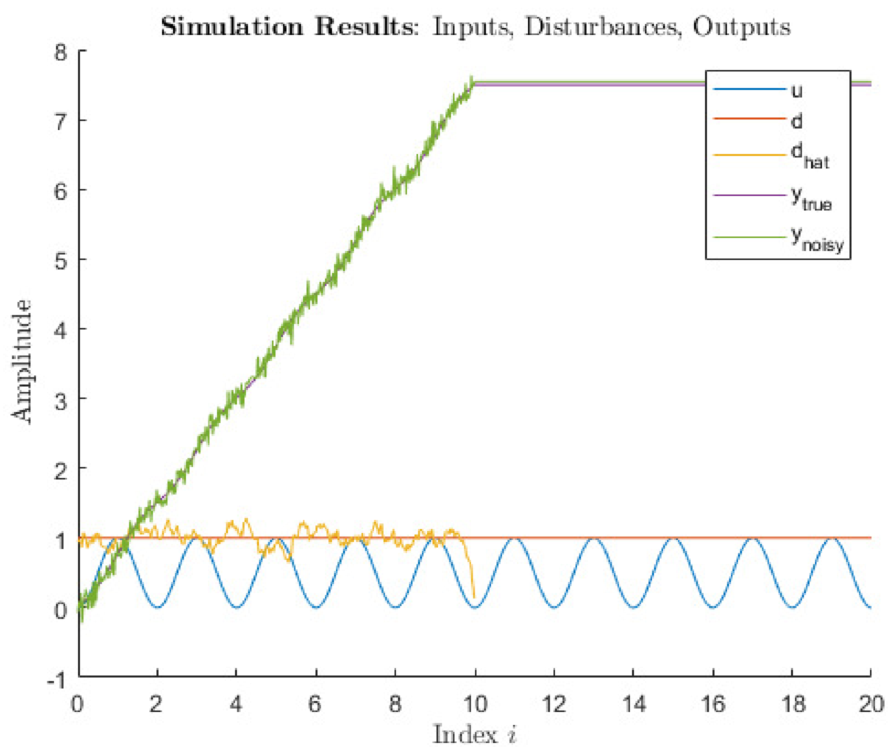
L–Curve (`q = 1`)



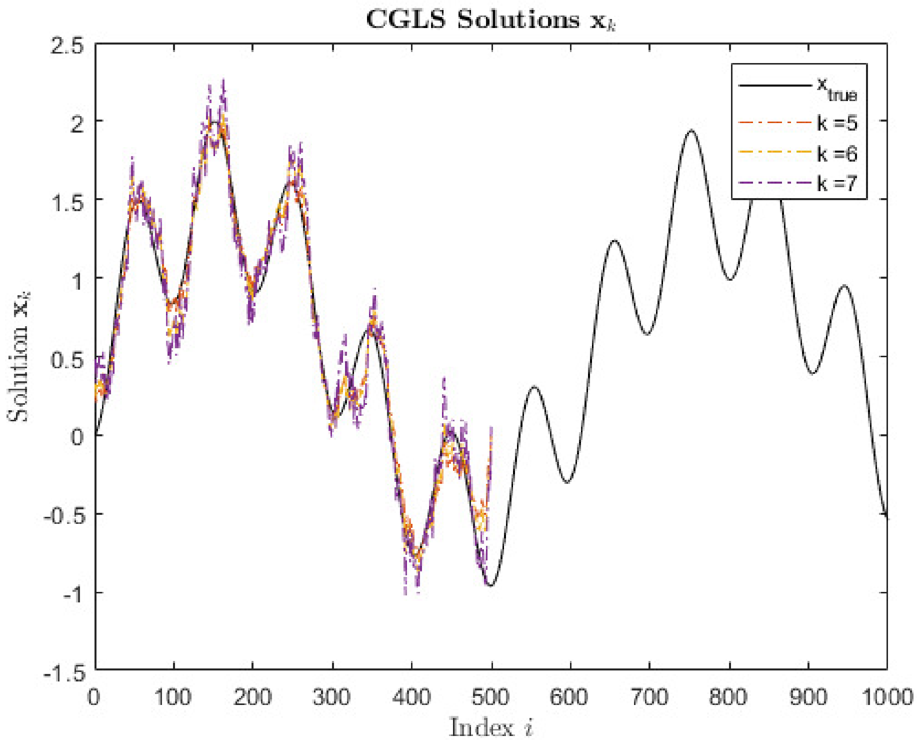
L–Curve (`q = 1`)



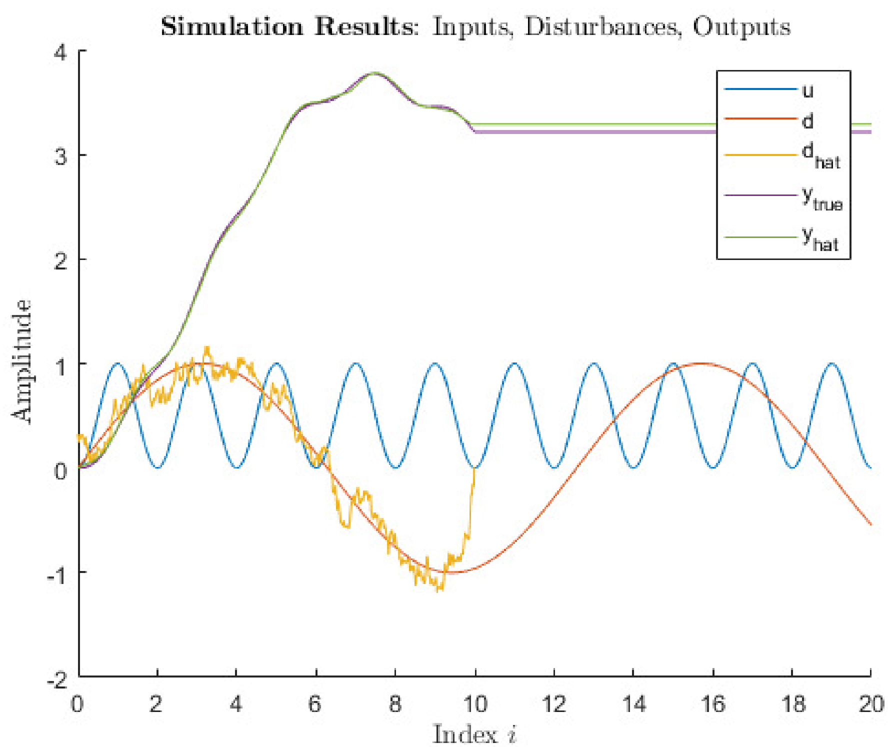
Constant Disturbance (q = 1)



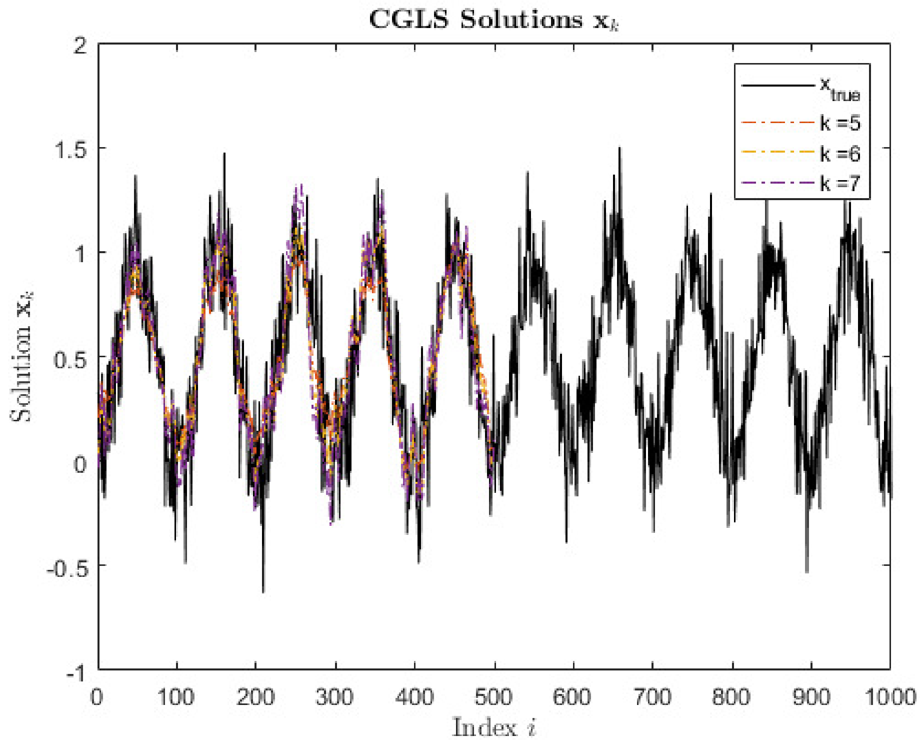
Constant Disturbance (q = 1)



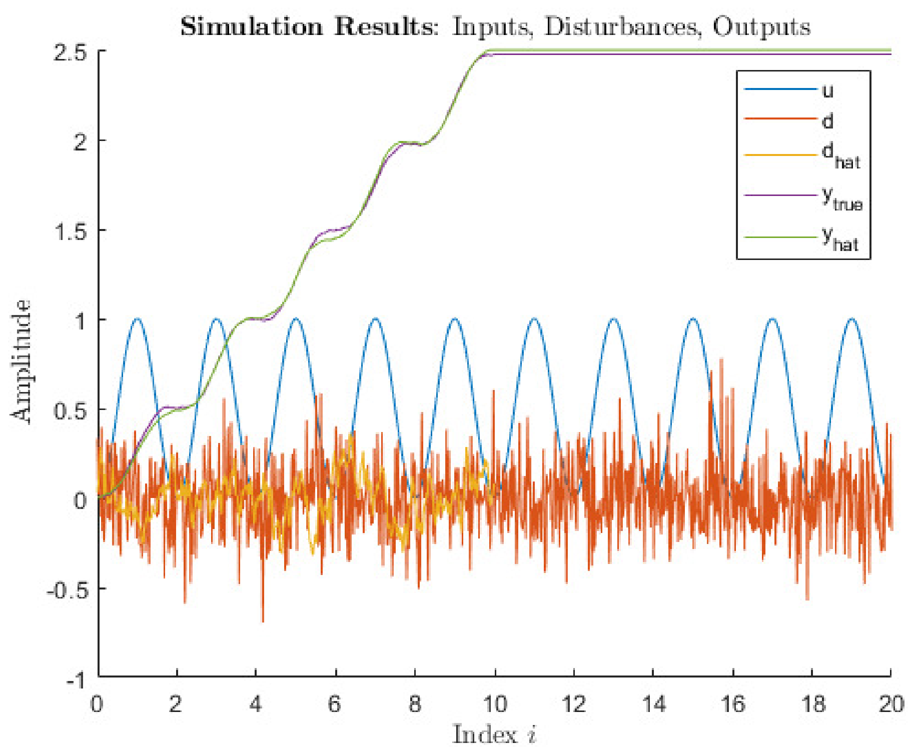
Sinusoidal Disturbance (`q = 2`)



Sinusoidal Disturbance (`q = 2`)



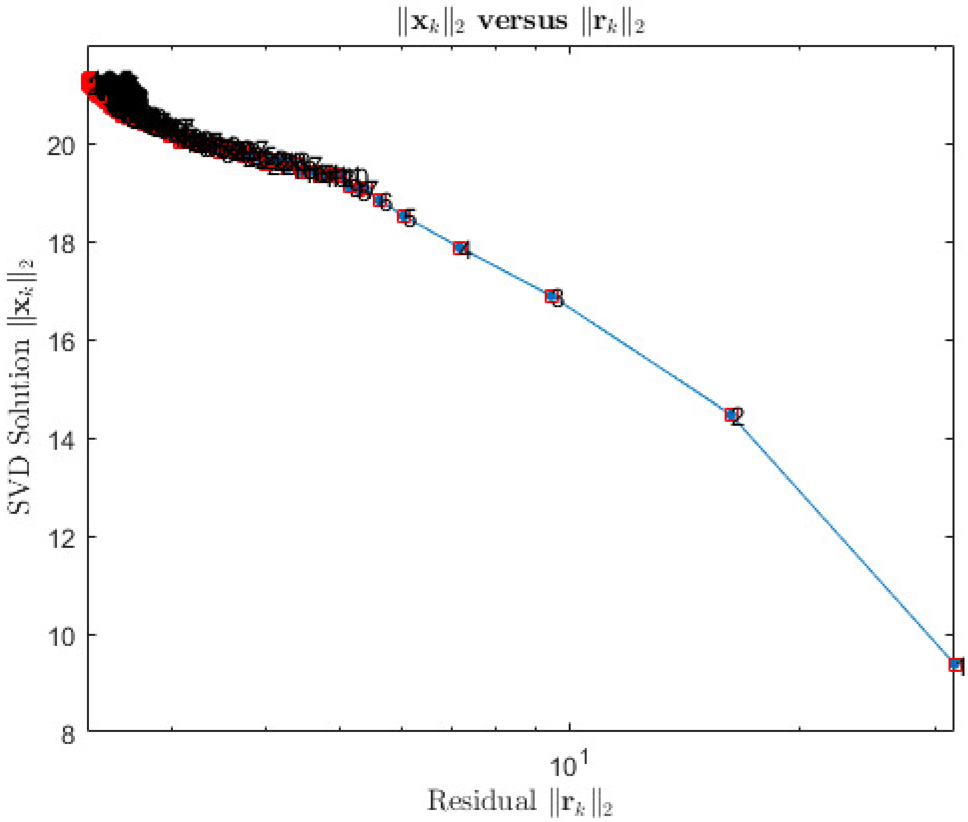
Gaussian Disturbance (q = 3)



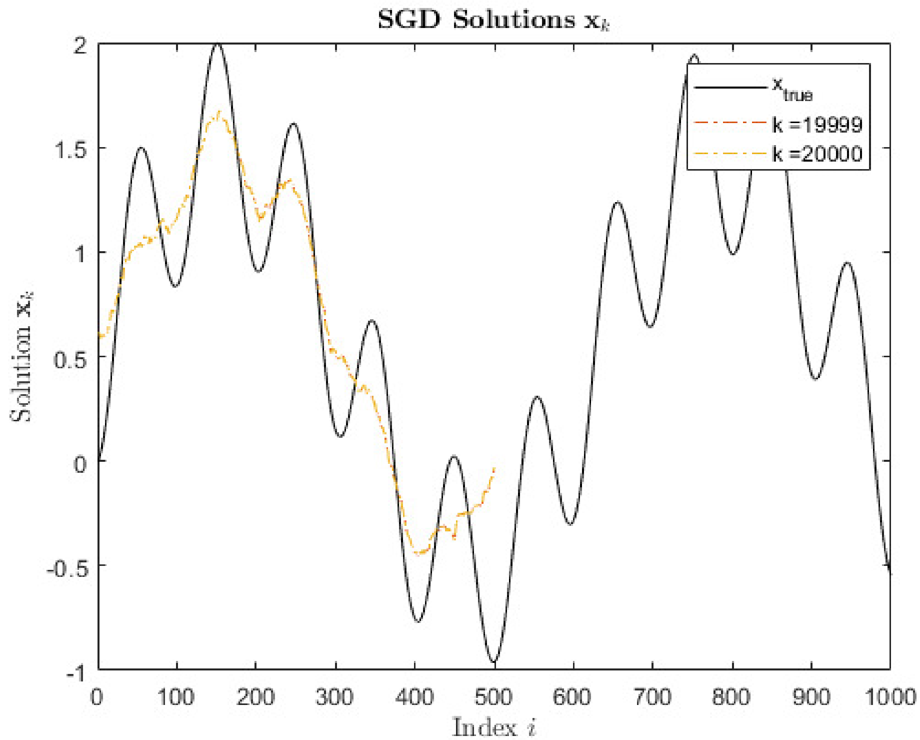
Gaussian Disturbance (q = 3)

Stochastic Gradient Descent

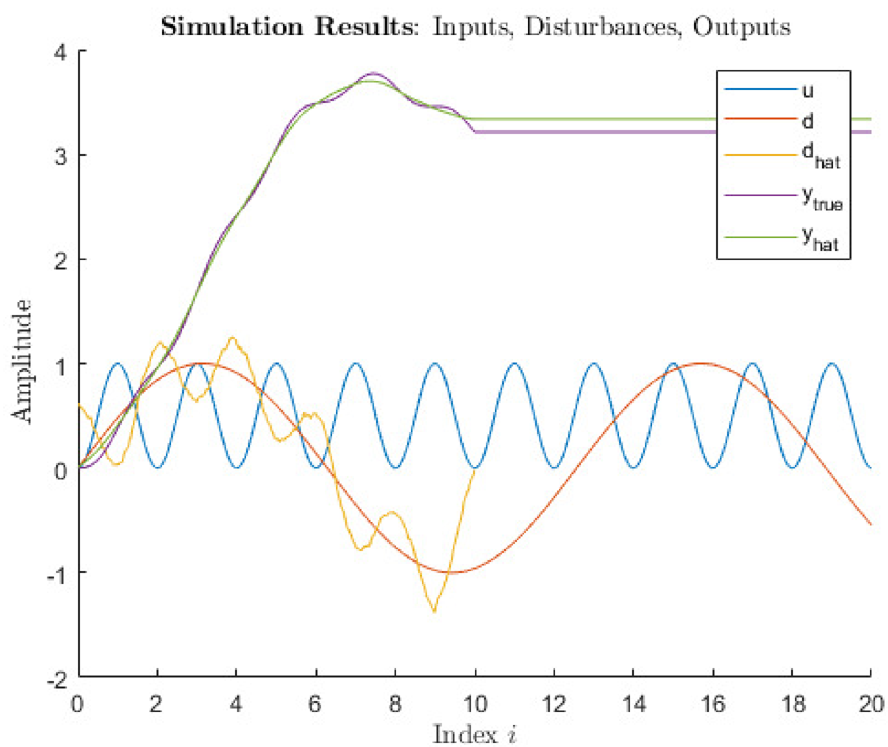
The following results are displayed for the ‘best’ SGD solutions.



L–Curve (`q = 1`)



Constant Disturbance (q = 1)



Constant Disturbance (q = 1)

Discussion

All methods were able to achieve some sense of success. However, they can be clearly ranked. For this problem, overall, CGLS performed best, and SGD performed worst. The nuances and details of the comparison are discussed further below.

Performance

CGLS reached its best solution is at most 8 iterations. This is in addition to its advantage of being an iterative scheme. It is, by far, the fastest solution among the three.

The Tikhonov SVD implementation, due to the nature of the factorization, scales badly with input size. Although the performance was acceptable, it was clearly inferior to CGLS.

SGD took between 10,000 and 20,000 iterations for acceptable results. It is much slower and is ranked last in terms of performance.

Accuracy

The CGLS and Tikhonov 'best' solutions were comparable in accuracy. The Tikhonov solution did result in slightly better estimated outputs $\hat{\mathbf{y}}$, but the difference is not too important. SGD yielded a much inferior input recovery, and due to its random nature, its solution contained some unwanted oscillatory behavior.

Conclusion

Overall, considering the simplicity, speed, and quality of CGLS, it is deemed the best method for this problem. The second-best is Tikhonov regularization. Last and least, SGD, while it worked, did not compare well, particularly for this problem.

Remarks

It was a bit surprising to see the formulation perform so well for a Gaussian disturbance. It is highly unintuitive to see a Gaussian signal estimated by another Gaussian signal *successfully*, yet the estimated output was good enough.

Working around the dimensional difference in the input and output spaces was not as straightforward as one would expect. While this did not necessarily worsen the ill-posedness, it did make it harder to recover the larger input from a smaller output.

For future work, it would be interesting to employ an inverse solver in a full control system.

References

[1] Chen, W. H., Yang, J., Guo, L., & Li, S. (2016). Disturbance-Observer-Based Control and Related Methods - An Overview. IEEE Transactions on Industrial Electronics. https://doi.org/10.1109/TIE.2015.2478397

[2] Disturbance Observer-Based Control: Methods and Applications. (2015). IEEE Control Systems. https://doi.org/10.1109/mcs.2015.2408011

[3] A. Radke and Zhiqiang Gao. (2006). A Survey of State and Disturbance Observers for Practitioners. American Control Conference, Minneapolis, MN, 2006. https://doi.org/10.1109/ACC.2006.1657545