BRIGHAM YOUNG UNIVERSITY, DEPARTMENT OF PHYSICS AND ASTRONOMY

Standalone Solver for Bayesian Compressive Sensing

Conrad W Rosenbrock

February 3, 2015

1 INTRODUCTION

In many physical systems, the number of experimental measurements available for fitting a mathematical model is severely constrained by time or cost. If the model relies on a fit in an orthonormal basis, a linear system of equations develops of the form:

$$\mathbb{A}\mathbf{j} = \mathbf{y}, \tag{1.1}$$

where $\mathbf{y}$ represents the experimental measurements that the model is being fit t; $\mathbb{A}$ is an $M \times N$ matrix of basis function evaluations for each measurement in $\mathbf{y}$; and $\mathbf{j}$ is the solution vector, or fit to the model. In cases where $M < N$, the system is called underdetermined and has an infinite number of solutions (since any vector in the kernel of $\mathbb{A}$ can be added to the solution vector $\mathbf{j}$ without effect). The question that Compressive Sensing (CS) [3] answers is "which of these infinite solutions is the best one, and how do I find it?".

A typical method for fitting functions to experimental values is to choose a set of coefficients $\mathbf{j}$ that minimizes the ℓ_2 norm of the predicted measurements relative to the true value $\mathbf{y}$:

$$\mathbf{j}_{\ell_2} = \underset{\mathbf{j}}{\operatorname{argmin}} \left\{ \epsilon : ||\mathbb{A}\mathbf{j} - \mathbf{y}||_2 < \epsilon \right\} = \underset{\mathbf{j}}{\operatorname{argmin}} \left\{ \epsilon : \left(\sum_i |\mathbb{A}\mathbf{j} - \mathbf{y}|_i^2 \right)^{\frac{1}{2}} < \epsilon \right\} \tag{1.2}$$

where ϵ is a finite tolerance on the desired error in the fit. In compressive sensing, a solution is found using the ℓ_1 norm instead:

$$\mathbf{j}_{\mathrm{CS}} = \underset{\mathbf{j}}{\operatorname{argmin}} \left\{ \left|\left| \mathbf{j} \right|\right|_1 : \left|\left| \mathbb{A}\mathbf{j} - \mathbf{y} \right|\right|_2 < \epsilon \right\}. \tag{1.3}$$

An additional constraint for using CS is that the solution needs to be sparse [2] in $\mathbf{j}$, meaning that there should be many components that are close to zero and only a few significant ones. If the representation basis is chosen appropriately, physical systems ought to be able to be cast within the constraints of CS.

The standalone solver assumes that you have collected several sets of data, with each set having:

1. an experimental measurement of some sort that is consider to be the true value. It is called y in the code.
2. evalutions of the function you are trying to fit for each basis function in your model. The final model will be a linear combination of these basis functions.

Each set of a measurement and the basis function values corresponding to that measurement are called a *data set* in the code. The rows of the $\mathbb{A}$ matrix (called Π in the code) represent distinct data sets. The columns are the basis function evaluations for each measurement.
In order to use this standalone solver, all you need initially is:

1. the $M \times N$ Π matrix described above.
2. the measurement vector $\mathbf{y}$ of length M.

1.1 Mathematical Constraints

One important constraint for the system is that the individual measurements have to satisfy an incoherence property (discussed in the Candés review paper [3, 2]). It is satisfied if your measurements are independent and identically distributed (IID) in the sense of a normal distribution. If you have an existing system, you can use the *bcs.choose_n_random_sets()* included in the *bcs.f90* module to choose a subset of the data sets in the system that are most closely IID. The routine returns a list of integers representing the indices of the systems (rows in Π, components of $\mathbf{y}$) that should be used in the other BCS fitting routines.

See the bibliography for references to a more rigorous discussion [5] of the constraints for CS and the mathematical concepts and proofs behind it. This write-up only serves as a quick guide to getting started. If you are doing anything serious, you should familiarize yourself with the constraints before using the code as a black-box.

1.2 Sparsity Enhancement

If you have constraints on the number of basis functions in your solution, you can use a reweighted ℓ_1 minimization routine with custom penalty functions that enhance the sparsity of the solution. Because of the complexity of certain models, dropping coefficients from

the model is not a simple choice: if two j_i values in the solution vector have similar values, how do you know which one will give the best fit?

The penalty functions improve sparsity as discussed in Candés paper; we include here some new penalty functions not originally introduced by Candés [4], some of which are related to transfer functions from machine learning and neural networks. As an example of the sparsity enhancing effects of each function, consider this list for a real-world problem that we solved using this algorithm.

- **No Penalty** $\left|\left|\mathbf{j}\right|\right|_1 = 120$, $\Delta_{\text{err}} = 8.3$.
- **logsum** $\left|\left|\mathbf{j}\right|\right|_1 = 48$, $\Delta_{\text{err}} = 8.3$.
- **arctan** $\left|\left|\mathbf{j}\right|\right|_1 = 32$, $\Delta_{\text{err}} = 8.3$.
- **quarti** $\left|\left|\mathbf{j}\right|\right|_1 = 22$, $\Delta_{\text{err}} = 8.3$.
- **hexics** $\left|\left|\mathbf{j}\right|\right|_1 = 13$, $\Delta_{\text{err}} = 8.3$.
- **octics** $\left|\left|\mathbf{j}\right|\right|_1 = 8$, $\Delta_{\text{err}} = 8.3$.
- **logsig** $\left|\left|\mathbf{j}\right|\right|_1 = 2$, $\Delta_{\text{err}} = 8.3$.

2 Parameter Descriptions

In this section, we briefly present the few tunable parameters that you can set, which affect the behavior of the solution vector. For a given set of parameters, the solution is deterministic. Roughly speaking, the algorithms tries to maximize the likelihood function (see the Babacan [1] and Tipping [6] papers) by choosing basis functions to add (i.e. j_i values to include in the sparse solution vector). As basis functions are added, removed or have their coefficients re-estimated, the maximum likelihood function of our model changes (as described in the papers).

- σ^2 represents the initial noise variance in the data. If you don't specify it, the value is calculated as $\left(\langle \mathbf{y}^2 \rangle - \langle \mathbf{y} \rangle^2\right)/100$.
- η is a threshold for stopping the iterations in the algorithm. If the change between the most recent iteration's $|\Delta ML|_{\text{latest}} < \eta |\Delta ML|_{\text{best}}$, then the iterations cease. Here $|\Delta ML|_{\text{best}}$ is the maximum gain in maximum likelihood achieved over the all iterations so far. Default value is 10^{-8}
- j_{cutoff} is the smallest value any j_i can have and still be included in the model. It is used only when the sparsity-enhancing reweighting routines are used and affects the weighting matrix. Default value is 10^{-3}.
- *penalty* is the name of one of the penalty functions described in the previous section for use in connection with the reweighting algorithm.

REFERENCES

[1] BABACAN, S. D., MOLINA, R., AND KATSAGGELOS, A. K. Bayesian Compressive Sensing Using Laplace Priors. *Ieee Transactions on Image Processing 19*, 1 (Jan. 2010), 53–63.

[2] CANDES, E., AND ROMBERG, J. Sparsity and Incoherence in Compressive Sampling. *arXiv.org* (Nov. 2006).

[3] CANDÈS, E. J., AND WAKIN, M. B. An Introduction To Compressive Sampling. *IEEE Signal Processing Magazine 25*, 2 (2008), 21–30.

[4] CANDÈS, E. J., WAKIN, M. B., AND BOYD, S. P. Enhancing Sparsity by Reweighted âĎŞ 1 Minimization. *Journal of Fourier Analysis and Applications 14*, 5-6 (Oct. 2008), 877–905.

[5] JI, S., XUE, Y., AND CARIN, L. Bayesian compressive sensing. *Ieee Transactions on Signal Processing 56*, 6 (June 2008), 2346–2356.

[6] TIPPING, M., AND FAUL, A. Fast Marginal Likelihood Maximization for Sparse Bayesian Models.