Support vector regression with non-smooth optimisation: an application of ADMM and FISTA using wavelet coefficients

Minh H. Nguyen

Department of Computer Science, UCL

20 May 2019

"Don't give yourselves to these unnatural men - machine men with machine minds and machine hearts!"

Charlie Chaplin, *The Great Dictator*

Abstract

In this paper, I use wavelet coefficients of the lagged 10-year US Treasury yield as features for a non-parametric support vector regression (SVR). Weights are obtained via optimisation over ℓ_1-norm loss functions and regularisers using methods in non-smooth and distributed optimisation. The primal problem is solved with alternating direction method of multipliers (ADMM) and the dual problem with the Gaussian kernel is solved via the fast iterative shrinkage-thresholding algorithm (FISTA). The primal optimisation yields a prediction function that is much more robust out of sample.

1 Introduction

The wavelet transform overcomes non-stationarity in financial time series with a decomposition at different desired scales to produce a good representation of the underlying structure of a stochastic process. To show that a true data generating process can be further circumvented, I employ support vector regression (SVR) to fit the wavelet coefficients onto the training set without parametric prescriptions on the data. SVR is an extension of a well-known and widely celebrated method in machine learning called support vector machines introduced by Cortes and Vapnik [7].

The paper's contribution is a comparison of different ways to attack the ℓ_1 and general non-smooth programmes that result from the derivation of the SVR solution. The paper explores various aspects of accuracy and optimisation difficulty differences between ADMM and FISTA, the standard and the kernel method as well as the primal and dual problem. The second section of this paper is dedicated to the dataset and its economic motivation, the third section gives a short overview of the basic idea behind wavelet decomposition and the fourth section goes over the optimisation algorithm that motivates the paper. The paper wraps up with a presentation of the optimisation result applied on the cross-validation set.

2 Data and Motivation

One of the most important concepts in asset pricing is the risk-free rate that the investor always has access to as a funding tool to price tradeable assets. A notable example of its uses is the Black-Scholes-Merton option pricing model that won Black and Scholes [3] the Nobel prize in economics in 1997. An example call price is given by the Feynman-Kac solution to a parabolic PDE

$$\begin{aligned} V_C(S_T, r, \sigma \mid S_t) &= \mathrm{e}^{-r(T-t)}\mathbb{E}_t^{\mathbb{Q}}\Big[\max\big(S_T - K, 0 \mid S_t\big)\Big] \\ &= \mathrm{e}^{-r(T-t)}\int_{\mathbb{R}} \big(S_T - K\big)^+ \,\mathrm{d}\Phi_{S_T}\left[\frac{\log\frac{S_T}{S_t} - \big(r - \frac{\sigma^2}{2}\big)(T-t)}{\sigma\sqrt{T-t}}\right] \end{aligned} \tag{2.1}$$

where V is the time-discounted value of the option (that expires at T) at time t, conditioned on the underlying's current price S_t with strike price K, where the underlying's maturity price S_T is filtered from a geometric Brownian Motion. The $\mathbb{E}^{\mathbb{Q}}$ notation denotes the expectation of the expression under the risk-neutral measure, representing the option price by its equivalent $\mathbb{Q}$-martingale conditioned on r. That is, we live in the physical measure $\mathbb{P}$ but we price assets in $\mathbb{Q}$ to rule out arbitrage.

A general economics framework is that of an economic agent maximising utility u over today and tomorrow, discounted by the risk-free interest rate, which reduces to balancing payoffs between two time periods by

$$\begin{aligned} X_t = \mathbb{E}_t^{\mathbb{P}}\left[\beta\frac{u'(X_{t+1})}{u'(X_t)}X_{t+1}\right] &= \mathrm{e}^{-rt}\mathbb{E}_t^{\mathbb{Q}}\big[X_{t+1}\big] \\ \text{s.t. } \mathbb{E}_t^{\mathbb{P}}\left[\beta\frac{u'(X_{t+1})}{u'(X_t)}\right] &= \frac{1}{1+r} \end{aligned} \tag{2.2}$$

The interpretation of this statement is that the risk-free interest rate should imply what the consumer/investor expects her "hunger" differential between today and tomorrow to be. Thus the $\mathbb{Q}$ measure relies heavily on the risk-free rate which incorporates prices of different states to enforce optimal decision making using what financial economists call Arrow-Debreu [1] securities. Note how r is constant in the standard model, implying that it is deterministic. But how deterministic is it in actual data and can we make non-trivial predictions for its future value?

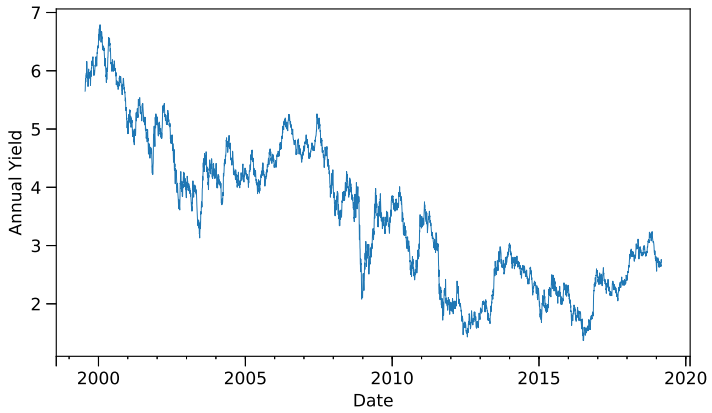


Figure 1: 10-Year Treasury yield

There are several interest rates that can be considered risk-free, but the most common choice in economics literature is a US government Treasury bond yield. I collected daily 10-year Treasury

rate from the St Louis Federal Reserve Economic Data website [10]. The 10-year Treasury note is noteworthy because is less volatile than the standard short-run risk-free Treasury bill rates and still holds information about long term growth outlook. The dataset spans from 20$^{\text{th}}$ July 1999 to 14$^{\text{th}}$ May 2019.

3 Support Vector Regression

Let $\mathbf{X} \in \mathbb{R}^{n \times m}$ be a matrix of m features and n observations with their corresponding output values $\mathbf{Y} \in \mathbb{R}^n$. Support vector machine finds a set of weights $\mathbf{w} \in \mathbb{R}^m$ that maximises the distance $2/\|\mathbf{w}\|_2$ between the positive and negative output values $\mathbf{Y}_i \in \{1, -1\}$. The pair of opposing classes' observations that have the smallest Euclidian distance in $\mathbb{R}^m$ form the support vectors and the function $f_\tau : \mathbb{R}^m \to \{1, -1\}$ supplies the prediction based on a specified threshold τ.

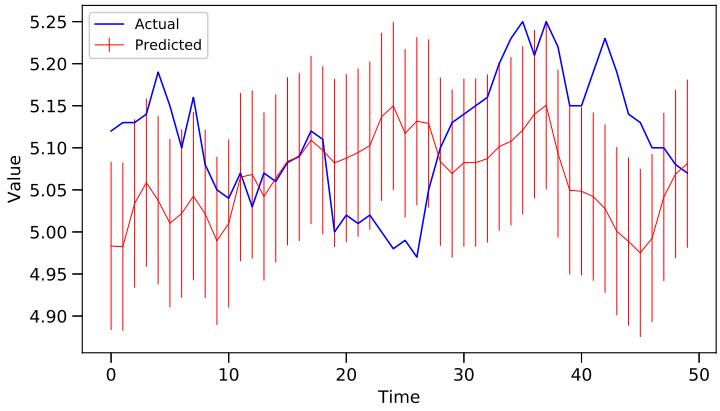


Figure 2: SVR prediction ϵ-tube ($\epsilon = 0.1$)

By letting the output support span the entire real line, $\mathbf{Y}_i \in \mathbb{R}$, we get a regression form for the prediction function $f : \mathbb{R}^m \to \mathbb{R}$ such that

$$f(\mathbf{X}_i) = \langle \mathbf{X_i}, \mathbf{w} \rangle + b \tag{3.1}$$

is allowed to vary within $\pm\epsilon$ from the true output $\mathbf{Y}_i$. By penalising all observations that violate the thresholds $\pm\epsilon$ we get the form of the ϵ-tube SVR, which is visualised in figure 2. Denoting ξ^+ and ξ^- as positive and negative violations of the ϵ-tube respectively, the optimisation problem reads

$$\begin{aligned} &\min_{\mathbf{w},b} \left\{ \frac{1}{2}\|\mathbf{w}\|_2^2 + C \sum_{i=1}^{n} (\xi_i^+ + \xi_i^-) \right\} \\ &\text{s.t.} \begin{cases} \langle \mathbf{X}_i, \mathbf{w} \rangle + b - \mathbf{Y}_i \leq \epsilon + \xi^+ \\ \mathbf{Y}_i - \langle \mathbf{X}_i, \mathbf{w} \rangle - b \leq \epsilon + \xi^- \\ \xi_i^+, \xi_i^- \geq 0 \quad \forall i = 1, ..., n \end{cases} \end{aligned} \tag{3.2}$$

which is interpreted as the sparsest predictor weight vector that minimises deviations from the ϵ-tube.

Note that (3.2) is equivalent to solving a smooth quadratic term and a non-smooth hinge loss that has two non-differentiable points at $-\epsilon$ and ϵ, i.e.

$$\xi_i = \begin{cases} 0 & \text{if } \left|\langle \mathbf{X}_i, \mathbf{w} \rangle + b - \mathbf{Y}_i\right| \leq \epsilon \\ \left|\langle \mathbf{X}_i, \mathbf{w} \rangle + b - \mathbf{Y}_i\right| - \epsilon & \text{otherwise} \end{cases} \tag{3.3}$$

Other forms of penalty that are also considered in literature are richly documented in [18]. A popular method is to utilise the smooth Huber loss that approximates the hinge loss with quadratic interpolation to make the edges differentiable. I proceed to show in section 6 that the inclusion of a damping matrix is enough for convergence.

Rewriting (3.2) in the form of its Lagrangian, we arrive at the following primal expression

$$\begin{aligned}\mathcal{L}(\mathbf{w}, b) = \Bigg\{ \frac{1}{2}\|\mathbf{w}\|_2^2 + C\sum_{i=1}^{n}(\xi_i^+ + \xi_i^-) - \sum_{i=1}^{n}(\lambda_i^+\xi_i^+ + \lambda_i^-\xi_i^-) \\ + \sum_{i=1}^{n}\alpha_i^+\Big(\mathbf{Y}_i - \langle \mathbf{w}, \mathbf{X}_i\rangle - b - \epsilon - \xi_i^+\Big) + \sum_{i=1}^{n}\alpha_i^-\Big(\langle \mathbf{w}, \mathbf{X}_i\rangle + b - \mathbf{Y}_i - \epsilon - \xi_i^-\Big)\Bigg\}\end{aligned} \tag{3.4}$$

where $\alpha^{(+)}$'s and $\lambda^{(+)}$'s are the corresponding Lagrange multipliers[1]. Partial differentiating with respect to the primal variables, we get the following optimality conditions

$$\frac{\partial \mathcal{L}}{\partial b} = \sum_{i=1}^{n}\alpha_i^+ - \sum_{i=1}^{n}\alpha_i^- = \sum_{i=1}^{n}\big(\alpha_i^+ - \alpha_i^-\big) = 0 \tag{3.5}$$

$$\nabla_{\mathbf{w}}\mathcal{L} = \mathbf{w} - \sum_{i=1}^{n}\alpha_i^+\mathbf{X}_i + \sum_{i=1}^{n}\alpha_i^-\mathbf{X}_i = 0 \implies \mathbf{w}^* = \sum_{i=1}^{n}\big(\alpha_i^+ - \alpha_i^-\big)\mathbf{X}_i \tag{3.6}$$

$$\frac{\partial \mathcal{L}}{\partial \xi_i^{(+)}} = C - \alpha_i^{(+)} - \lambda^{(+)} = 0 \implies \lambda^{(+)} = C - \alpha_i^{(+)} \tag{3.7}$$

By substituting these optimality conditions back into (3.4) and conveniently eliminating all $\lambda_i^{(+)}$, we arrive at the dual function

$$\begin{aligned}g\big(\boldsymbol{\alpha}^{(+)}\big) = \inf \mathcal{L}(\mathbf{w}, b) = \Bigg\{ -\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\big(\alpha_i^+ - \alpha_i^-\big)\big(\alpha_j^+ - \alpha_j^-\big)\langle \mathbf{X}_i, \mathbf{X}_j\rangle \\ -\epsilon\sum_{i=1}^{n}\big(\alpha_i^+ + \alpha_i^-\big) + \sum_{i=1}^{n}\mathbf{Y}_i\big(\alpha_i^+ - \alpha_i^-\big)\Bigg\}\end{aligned} \tag{3.8}$$

Closing the duality gap then reduces to the dual problem, which can be reformulated from a maximisation problem to a minimisation problem:

$$\begin{aligned}\min \Bigg\{ \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\big(\alpha_i^+ - \alpha_i^-\big)\big(\alpha_i^+ - \alpha_i^-\big)\langle \mathbf{X}_i, \mathbf{X}_j\rangle + \epsilon\sum_{i=1}^{n}\big(\alpha_i^+ + \alpha_i^-\big) - \sum_{i=1}^{n}\mathbf{Y}_i\big(\alpha_i^+ - \alpha_i^-\big)\Bigg\} \\ \text{s.t. } \begin{cases}\sum_{i=1}^{n}\big(\alpha_i^+ - \alpha_i^-\big) = 0 \\ \alpha_i^{(+)} \in [0, C]\end{cases}\end{aligned} \tag{3.9}$$

Because of (3.6), we get an indirect expression for the weights and bias, given by the affine function

$$f(\mathbf{X}_u) = \sum_{i}^{n}\big(\alpha_i^+ - \alpha_i^-\big)\langle \mathbf{X}_i, \mathbf{X}_u\rangle + b \tag{3.10}$$

The majority of optimisation algorithms work with the dual problem to exploit Mercer's [13] theorem and the reproducing kernel Hilbert space property.

[1]The notation $^{(+)}$ denotes application to both positive and negative values.

Theorem 1 (Mercer)**.** *Let $X \subset \mathbb{R}^n$ be a measure space, μ a strictly positive Borel measure on X, $\{a_i\}_{i=1}^n$,$\{x_i\}_{i=1}^n \in \mathbb{R}^n$ finite sequences, $K : X \times X \to \mathbb{R}$ a positive semi-definite kernel on X*

$$\sum_{i,j=1}^{n} a_i a_j K(x_i, x_j) \geq 0 \tag{MT1}$$

that is square-integrable

$$\int_{X\times X} K(x,y)^2 \,\mathrm{d}\mu(x)\,\mathrm{d}\mu(y) < \infty \tag{MT2}$$

then it has an eigen-decomposition

$$K(\mathbf{x},\mathbf{y}) = \sum_{l=1}^{\infty} \lambda_l \psi_l(\mathbf{x})\psi_l(\mathbf{y}) = \big\langle \phi(\mathbf{x}), \phi(\mathbf{y}) \big\rangle \tag{MT3}$$

where the sum converges for each pair $(\mathbf{x},\mathbf{y}) \in X \times X$ and uniformly on each compact subset of X. The $L^2(X)$-normalised eigenfunctions $\{\phi_i\}_{i=1}^{\infty}$ form an orthonormal basis of the Hilbert space $L^2(X)$. We say that K is Mercer.

As such, the inner product in f which is given by the covariance matrix of $\mathbf{X}$ (if centred) can be replaced with any kernel that has a representation that satisfies (MT3), i.e. $\langle \mathbf{x}, \mathbf{y} \rangle \to \langle \phi(\mathbf{x}), \phi(\mathbf{y}) \rangle$. This is known as the kernel "trick" in learning literature since it transforms data from their original domain into higher dimensional space that may induce linear separability, where the transformed coordinates through ϕ do not have to be calculated explicitly. The Gaussian kernel used in the dual problem is defined as

$$K_{Gauss}(\mathbf{x},\mathbf{y}) = \exp\left(\frac{-a\|\mathbf{x}-\mathbf{y}\|_2^2}{2\sigma^2} + c\right) \tag{3.11}$$

with $a = 1, \sigma = 1, c = 0$. I chose to solve both the primal and the dual problem to highlight their prediction differences, but one can solve the kernel problem in the primal as well [5].

4 Wavelet Decomposition

Fourier analysis suffers from the lack of time localisation of frequency information and vice versa. The phenomenon is known to physicists and engineers as the Heisenberg principle of uncertainty. Wavelet analysis addresses this problem with a scheme which deliberately returns poor time localisation of low frequencies and good time localisation of high frequencies. The idea is that low frequencies propagate throughout the signal and high frequencies are mostly localised.

Skipping the full theory that can be found in [11], a discrete signal can be decomposed into a linear combination of orthonormal bases by

$$\big\{f[t]\big\}_{t=1}^{\infty} = \sum_{k\in\mathbb{Z}} \big\langle f, \mathrm{T}_k\phi \big\rangle \mathrm{T}_k\phi + \sum_{j=1}^{\infty}\sum_{k\in\mathbb{Z}} \big\langle f, \mathrm{D}^j \circ \mathrm{T}_k\psi \big\rangle D^j \circ \mathrm{T}_k\psi \tag{4.1}$$

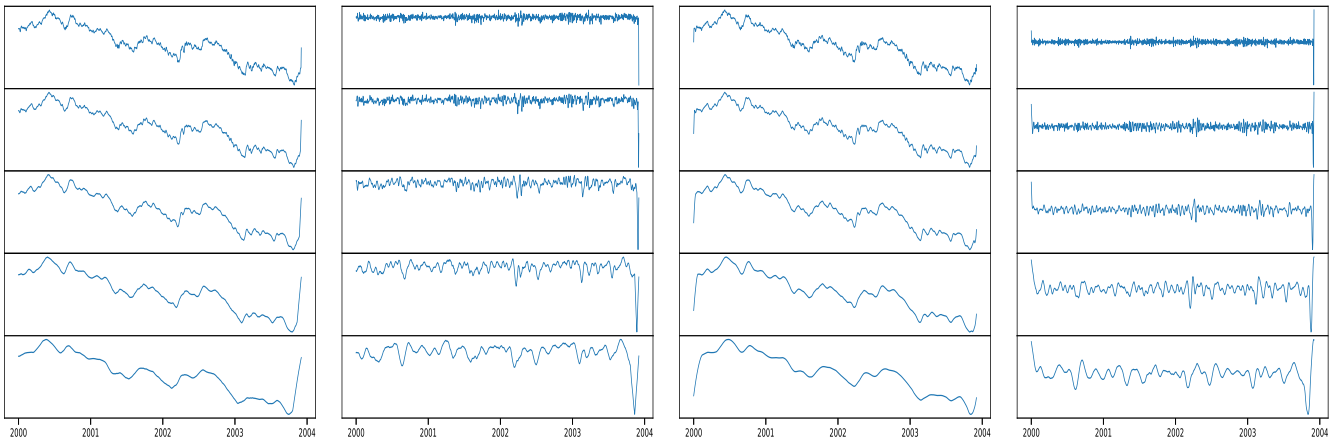
where T and D denote the translation and dilation operator respectively. The scaling function ϕ, also called the father wavelet, plays the role of a low-pass filter at scale j and ψ, called the mother wavelet, is the high pass filter at scale j.

In practice, D is always 2 and all the functions have finite support so that the down sampling

process halves the number of samples and enlarges the "resolution" window by a factor of two at each discrete scale. Using standard notations, I define the series of signal observations $\{f[t]\}_{t=1}^{N}$ as vector $\mathbf{X}$ and rewrite (4.1) as

$$\mathbf{X} = \mathcal{V}_J^{\mathrm{T}}\mathbf{V}_J + \sum_{j=1}^{J} \mathcal{W}_j^{\mathrm{T}}\mathbf{W}_j \tag{4.2}$$

where, at scale J, $\mathcal{W}_j$ contains the dilated and translated wavelets at the below and current scales, $\mathcal{V}_J$ contains the approximating low-pass filter of current scale, $\mathbf{X}_j$ contains the high-frequency coefficients at below and current scale and $\mathbf{V}_J$ contains the approximating or trend coefficients at current scale.



(a) Haar Approximations (b) Haar Details (c) db2 Approximations (d) db2 Details

Figure 3: Haar and db2 SWT of the training set output

The downside of the DWT is that coefficients are not invariant under translation due to the loss of series alignment at each downsampling scale. I instead use the stationary wavelet transform (SWT), which retains all coefficients and upsamples at each scale by padding zeros to retain the full length N number of coefficients at each scale. The SWT, however, forfeits the orthonormal bases due to the upsampling process, so the energy is no longer preserved as in the case with the DWT due to the redundancy. From [16], we instead have

$$\|\mathbf{X}\|_2^2 = \sum_{j=1}^{J} \left\| T_J^{\nu(G)}\tilde{\mathbf{V}}_J \right\|_2^2 + \left\| T_j^{\nu(H)}\tilde{\mathbf{W}}_j \right\|_2^2 \tag{4.3}$$

where $\nu(H)$ and $\nu(G)$ represent the circular shifts of the mother and father wavelet respectively and taking the mean. The discrete wavelet tested is the Haar wavelet

$$\begin{aligned} \psi_{Haar}[t] &= \begin{cases} 1, & 0 \le t < \frac{1}{2} \\ -1, & \frac{1}{2} \le t < 1 \\ 0 & \text{otherwise.} \end{cases} \\ \phi_{Haar}[t] &= \begin{cases} 1, & 0 \le t < 1 \\ 0 & \text{otherwise.} \end{cases} \end{aligned} \tag{4.4}$$

The continuous wavelet tested is the Daubechies-2 (db2) wavelet, which does not have analytical forms but we can use the Fourier representations of their filters

$$\begin{aligned}\hat{\psi}_{db2}[\xi] &= \prod_{k=1}^{\infty} \frac{\hat{H}(2^{-k}\xi)}{\sqrt{2}} \\ \hat{\phi}_{db2}[\xi] &= \frac{1}{\sqrt{2}}\hat{G}\left(\frac{\xi}{2}\right)\hat{\phi}\left(\frac{\xi}{2}\right)\end{aligned} \tag{4.5}$$

where $\hat{f}$ denotes the Fourier series of f

$$\hat{f}[\xi] = \sum_{k=0}^{n-1} f[k]\mathrm{e}^{-\mathrm{i}k\xi} \tag{4.6}$$

The inclusion of the Daubechies-2 (db2) wavelet is therefore to compare performances between a discrete and a continuous transform. Testing the 1^{st}, 4^{th} and 20^{th} lags, I get the corresponding 5-level deep SWT coefficients of both approximation and detail at each level. The training set is the first 1044 observations of the dataset (1024 for the SWT and 20 first observations removed due to the lags). The detail and final approximation coefficients are then used as the feature matrix for SVR, giving $n = 1024, m = 6$.

5 Algorithms

5.1 Alternating Direction of Multipliers

The motivation for ADMM is that it blends the separability of dual ascent algorithms with the fast convergence of the method of multipliers. Given a problem that is separable as in the dual ascent case, we can rewrite it in the canonical ADMM form

$$\begin{aligned}&\min f(\mathbf{x}) + g(\mathbf{y}) \\ &\text{s.t. } \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{y} = \mathbf{c}\end{aligned} \tag{5.1}$$

and its corresponding augmented Lagrangian is

$$\underset{\mathbf{x},\mathbf{y},\lambda,\rho}{\mathcal{L}} = f(\mathbf{x}) + g(\mathbf{y}) + \langle \lambda, \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{y} - \mathbf{b}\rangle + \frac{\rho}{2}\|\mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{y} - \mathbf{c}\|_2^2 \tag{5.2}$$

The general form of the iterations reads

$$\begin{aligned}\mathbf{x}^{k+1} &= \arg\min_{\mathbf{x}} \mathcal{L}(\mathbf{x}, \mathbf{y}^k, \lambda) \\ \mathbf{y}^{k+1} &= \arg\min_{\mathbf{y}} \mathcal{L}(\mathbf{x}^{k+1}, \mathbf{y}, \lambda) \\ \lambda^{k+1} &= \lambda^k + \rho(\mathbf{A}\mathbf{x}^{k+1} + \mathbf{B}\mathbf{y}^{k+1} - \mathbf{c})\end{aligned} \tag{5.3}$$

which separate the variables and update them sequentially conditioned on the previous variable at every iteration. The algorithm works well for large datasets that can be broken down into disjoint subsets.

By defining a residual variable $\mathbf{r} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{y} - \mathbf{c}$, we get a more useful scaled form of (5.2)

$$\begin{aligned}\underset{\mathbf{x},\mathbf{y},\lambda,\rho}{\mathcal{L}} &= f(\mathbf{x}) + g(\mathbf{y}) + \langle \lambda, \mathbf{r}\rangle + \frac{\rho}{2}\|\mathbf{r}\|_2^2 \\ &= f(\mathbf{x}) + g(\mathbf{y}) + \frac{\rho}{2}\left\|\mathbf{r} + \frac{1}{\rho}\lambda\right\|_2^2 + \frac{\rho}{2}\|\lambda\|_2^2 \\ \underset{\mathbf{x},\mathbf{y},\boldsymbol{\theta}}{\mathcal{L}} &= f(\mathbf{x}) + g(\mathbf{y}) + \frac{\rho}{2}\|\mathbf{r} + \boldsymbol{\theta}\|_2^2 + \frac{\rho}{2}\|\boldsymbol{\theta}\|_2^2\end{aligned} \tag{5.4}$$

where $\boldsymbol{\theta} = (1/\rho)\lambda$ is the scaled dual variable. The iterates of the scaled ADMM are thus given by

$$\begin{aligned}
\mathbf{x}^{k+1} &= \arg\min_{\mathbf{x}} \left\{ f(\mathbf{x}) + \frac{\rho}{2} \left\| \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{y}^k - \mathbf{c} + \boldsymbol{\theta}^k \right\|_2^2 \right\} \\
\mathbf{y}^{k+1} &= \arg\min_{\mathbf{y}} \left\{ g(\mathbf{y}) + \frac{\rho}{2} \left\| \mathbf{A}\mathbf{x}^{k+1} + \mathbf{B}\mathbf{y} - \mathbf{c} + \boldsymbol{\theta}^k \right\|_2^2 \right\} \\
\boldsymbol{\theta}^{k+1} &= \boldsymbol{\theta}^k + \mathbf{A}\mathbf{x}^{k+1} + \mathbf{B}\mathbf{y}^{k+1} - \mathbf{c}
\end{aligned} \tag{5.5}$$

and the goal here is to drive $\|\boldsymbol{\theta}\|_2$ to 0, which corresponds to no primal gap.

Because the problem is convex, the solution is unique and the algorithm converges under fairly mild assumptions. A complete convergence proof can be found in [4], but I will proceed to give a concise statement for concreteness.

Theorem 2. *Let $f : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ and $g : \mathbb{R}^m \to \mathbb{R} \cup \{+\infty\}$ be closed, proper and convex functions whose codomains span the extended real line $\iff$ the epigraph of f*

$$\text{epi } f = \left\{ (x, t) \in \mathbb{R}^n \times \mathbb{R} \mid f(x) \leq t \right\} \tag{5.6}$$

is a closed, non-empty convex set. Furthermore, assume that the augmented Lagrangian (5.4) has a saddle point, i.e. $\exists\, \mathbf{x}^*, \mathbf{y}^*, \boldsymbol{\theta}^*$ s.t.

$$\mathcal{L}(\mathbf{x}^*, \mathbf{y}^*, \boldsymbol{\theta}) \leq \mathcal{L}(\mathbf{x}^*, \mathbf{y}^*, \boldsymbol{\theta}^*) \leq \mathcal{L}(\mathbf{x}, \mathbf{y}, \boldsymbol{\theta}^*) \tag{5.7}$$

then by assumption 1, $\mathcal{L}(\mathbf{x}^*, \mathbf{y}^*, \boldsymbol{\theta}^*) < \infty$, $\mathbf{A}\mathbf{x}^* + \mathbf{B}\mathbf{y}^* = \mathbf{c}$ and $f(\mathbf{x}^*) < \infty, g(\mathbf{y}^*) < \infty$. This implies that $\boldsymbol{\theta}^*$ is dual optimal and strong duality holds.

The state-of-the-art algorithm to solve support vector problems is LIBSVM which solves the dual Karush-Kuhn-Tucker system in (3.9) for each pair of dual variables, but solving in the primal is gives the prediction weight a layer of insensitivity to outlier training observations.

The SVR primal optimisation problem from (3.2) can be reformulated as a non-smooth programme as follows

$$\min_{\mathbf{w}, b} \quad \left\{ \lambda \|\mathbf{w}\|_2^2 + \sum_{i=1}^{n} \max\left(0, |\langle \mathbf{w}, \mathbf{X}_i \rangle + b - \mathbf{Y}_i| - \epsilon\right) \right\} \tag{5.8}$$

where $\lambda = (1/2C)$. The sum in (5.8) is called hinge loss, which consists of n ϵ-insensitive loss functions. The first term is smooth and differentiable, whereas the hinge loss is non-smooth. Notice that we can remove b by adding a column of 1's to the $\mathbf{X}$ matrix, hence simplifying the expression. I define 2 diagonal matrices $\mathbf{Q}^U, \mathbf{Q}^L \in \mathbb{R}^{n \times n}$ with their elements given by

$$\mathbf{Q}^U_{ii} = \begin{cases} \langle \mathbf{w}, \mathbf{X}_i \rangle - \mathbf{Y}_i - \epsilon & \text{if } \langle \mathbf{w}, \mathbf{X}_i \rangle - \mathbf{Y}_i > \epsilon \\ 0 & \text{otherwise} \end{cases} \tag{5.9}$$

$$\mathbf{Q}^L_{ii} = \begin{cases} -\langle \mathbf{w}, \mathbf{X}_i \rangle + \mathbf{Y}_i - \epsilon & \text{if } \langle \mathbf{w}, \mathbf{X}_i \rangle - \mathbf{Y}_i < -\epsilon \\ 0 & \text{otherwise} \end{cases} \tag{5.10}$$

and these matrices would change at every iteration until the convergence criterion has been met, i.e. until they produce little penalty. I arrived at this choice of damping weight after some experimenting with different weights. The damped loss drives ϵ violations apart from each other, reducing likelihood

of the losses falling into non-smooth neighbourhoods upon initialisation. For a feasible solution the damped loss is equivalent to the hinge loss, which corresponds to setting the penalties to 1 instead. The new, slightly altered and unconstrained problem is then

$$\min_{\mathbf{w}} \quad \Big\{\lambda\|\mathbf{w}\|_2^2 + \mathbf{1}_n^{\mathrm{T}}\mathbf{Q}^U\big(\mathbf{X}\mathbf{w} - \mathbf{Y} - \epsilon\mathbf{1}_n\big) - \mathbf{1}_n^{\mathrm{T}}\mathbf{Q}^L\big(\mathbf{X}\mathbf{w} - \mathbf{Y} + \epsilon\mathbf{1}_n\big)\Big\} \tag{5.11}$$

where $\mathbf{1}_n \in \mathbb{R}^n$ is a vector of ones.

By dividing the dataset into p subsets, we can optimise each subset disjointly from the others and drive the solutions of the subsets toward a consensus after each iteration with the split variable as in (5.5). The set up is formulated as

$$\begin{gathered}\min_{\mathbf{w}_1,\ldots,\mathbf{w}_p} \quad \Big\{\lambda\|\mathbf{z}\|_2^2 + \sum_{j=1}^{p}\sum_{i=1}^{n_j}\Big[\mathbf{Q}_{j,ii}^U\big(\langle\mathbf{w}_j, \mathbf{X}_{j,i}\rangle - \mathbf{Y}_{j,i} - \epsilon\big) - \mathbf{Q}_{j,ii}^L\big(\langle\mathbf{w}_j, \mathbf{X}_{j,i}\rangle - \mathbf{Y}_{j,i} + \epsilon\big)\Big]\Big\} \\ \text{s.t. } \mathbf{w}_j = \mathbf{z}\ \forall\ j\end{gathered} \tag{5.12}$$

where the subscript j denotes the value of the variable belonging to subset j. The new total augmented Lagrangian is given by

$$\begin{aligned}\underset{\mathbf{w}}{\mathcal{L}} = \lambda\|\mathbf{z}\|_2^2 + \sum_{j=1}^{p}\Big\{&\mathbf{1}_{n_j}^{\mathrm{T}}\mathbf{Q}_j^U\big(\mathbf{X}_j\mathbf{w}_j - \mathbf{Y}_j - \epsilon\mathbf{1}_{n_j}\big) - \mathbf{1}_{n_j}^{\mathrm{T}}\mathbf{Q}_j^L\big(\mathbf{X}_j\mathbf{w}_j - \mathbf{Y}_j + \epsilon\mathbf{1}_{n_j}\big) \\ &+\langle\boldsymbol{\mu}, \mathbf{w}_j - \mathbf{z}\rangle + \frac{\rho}{2}\|\mathbf{w}_j - \mathbf{z}\|_2^2\Big\}\end{aligned} \tag{5.13}$$

where $\boldsymbol{\mu} \in \mathbb{R}^m$ is the vector of dual variables for the dual gap. Noting the disjointness between any $\mathbf{w}_j$ and $\mathbf{z}$, the new iterates' updates are given by

$$\begin{aligned}\mathbf{w}_j^{k+1} = \arg\min_{\mathbf{w}_j}\Big\{&\mathbf{1}_{n_j}^{\mathrm{T}}\mathbf{Q}_j^{U(k)}\big(\mathbf{X}_j\mathbf{w}_j - \mathbf{Y}_j - \epsilon\mathbf{1}_{n_j}\big) - \mathbf{1}_{n_j}^{\mathrm{T}}\mathbf{Q}_j^{L(k)}\big(\mathbf{X}_j\mathbf{w}_j - \mathbf{Y}_j + \epsilon\mathbf{1}_{n_j}\big) \\ &+ \langle\boldsymbol{\mu}_j, \mathbf{w}_j - \mathbf{z}^k\rangle + \frac{\rho}{2}\|\mathbf{w}_j - \mathbf{z}^k\|_2^2\Big\}\end{aligned} \tag{5.14}$$

$$\mathbf{z}^{k+1} = \arg\min_{\mathbf{z}}\Big\{\lambda\|\mathbf{z}\|_2^2 + \sum_{j=1}^{p}\Big(\langle\mathbf{z}, -\boldsymbol{\mu}_j\rangle + \frac{\rho}{2}\|\mathbf{w}_j^{k+1} - \mathbf{z}\|_2^2\Big)\Big\} \tag{5.15}$$

$$\boldsymbol{\mu}_j^{k+1} = \boldsymbol{\mu}_j^k + \rho\big(\mathbf{w}_j^{k+1} - \mathbf{z}^{k+1}\big) \tag{5.16}$$

where n_j is the number of observations subset j.

When the weights agree, (5.16) stops updating and this happens when the penalty terms in (5.14) and (5.15) vanish. These then act as a stopping criterion which is given in (5.20) [4]. Following (5.4), I let $\mathbf{t}_j = (1/\rho)\boldsymbol{\mu}_j$, rewrite (5.14) as

$$\begin{aligned}\mathbf{w}_j^{k+1} = \arg\min_{\mathbf{w}_j}\varphi = \arg\min_{\mathbf{w}_j}\Big\{&\mathbf{1}_{n_j}^{\mathrm{T}}\mathbf{Q}_j^{U(k+1)}\big(\mathbf{X}_j\mathbf{w}_j - \mathbf{Y}_j - \epsilon\mathbf{1}_{n_j}\big) \\ &-\mathbf{1}_{n_j}^{\mathrm{T}}\mathbf{Q}_j^{L(k+1)}\big(\mathbf{X}_j\mathbf{w}_j - \mathbf{Y}_j + \epsilon\mathbf{1}_{n_j}\big) + \frac{\rho}{2}\big\|\mathbf{w}_j - \mathbf{z}^k + \mathbf{t}_j^k\big\|_2^2\Big\}\end{aligned} \tag{5.17}$$

which is the scaled form, and obtain its gradient

$$\nabla_{\mathbf{w}_j} \varphi = \rho(\mathbf{w}_j - \mathbf{z}_j - \mathbf{t}) + \mathbf{1}_{n_j}^{\mathrm{T}}\big(\mathbf{Q}_j^U - \mathbf{Q}_j^L\big)\mathbf{X}_j \tag{5.18}$$

To get an analytical solution for (5.15), we minimise over the sample average by setting its gradient to zero and perform some algebra to get

$$\mathbf{z}^{k+1} = \frac{p\rho}{2\lambda + p\rho}\Big(\overline{\mathbf{w}^{k+1}} + \overline{\mathbf{t}^k}\Big) \tag{5.19}$$

where $\overline{\mathbf{t}^k}$ denotes the iteration's average value of $\mathbf{t}^k$. Substituting $\rho\mathbf{t}_j = \boldsymbol{\mu}_j$ into (5.16), we get an individual slack variable update of the form

$$\mathbf{t}_j^{k+1} = \mathbf{t}_j^k + \mathbf{w}_j^{k+1} - \mathbf{z}^{k+1} \tag{5.20}$$

Algorithm 1: Distributed ADMM for SVR

Initialise $p, \gamma, C \to \lambda = (1/2C)$
$\mathbf{w_j}, \mathbf{t}_j, \mathbf{z} = 0\mathbf{1}_m \; \forall \; j = 1, ..., p$
$abstol = 1\mathrm{e}^{-4}$
$reltol = 1\mathrm{e}^{-5}$
while *not converged* **do**
  for *j=1,...,p* **do**
    $\mathbf{w}_j^{k+1}$ = Use Conjugate Gradient to solve (5.17)
  end for
  Compute average $\overline{\mathbf{w}^{k+1}}$
  $\mathbf{w}^{k+1} = \alpha\mathbf{w}^{k+1} + (1-\alpha)\mathbf{z}^k$ | (Over-relaxation [4])
  $\mathbf{z}^{k+1} = \frac{p\rho}{2\lambda+p\rho}\Big(\overline{\mathbf{w}^{k+1}} + \overline{\mathbf{t}^k}\Big)$
  $\mathbf{t}_j^{k+1} = \mathbf{t}_j^k + \mathbf{w}_j^{k+1} - \mathbf{z}^{k+1}$
end while
return $\mathbf{w}^{k+1}$;

I set the stopping condition as advised in Boyd et. al [4], where they set out two initial tolerance levels called absolute and relative tolerance. Every single iteration updates a new value for each tolerance level depending on the last iteration's accuracy

$$\begin{aligned} \epsilon_{primal}^{k+1} &= \sqrt{m}\epsilon_{abs} + \epsilon_{rel}\max\big(\|\mathbf{w}^{k+1}\|_2, \|\mathbf{z}^{k+1}\|_2\big) \\ \epsilon_{dual}^{k+1} &= \sqrt{m}\epsilon_{abs} + \epsilon_{rel}\|\rho\mathbf{t}^{k+1}\|_2 \end{aligned} \tag{5.21}$$

Empirical observations of CG's stability naturally led me to another stopping criterion based on the hinge loss in (5.15), which stops the loop and returns the solution if

$$\sum_{i=1}^{n} \max\big(0, |\langle \mathbf{w}, \mathbf{X}_i\rangle + b - \mathbf{Y}_i| - \epsilon\big) \leq \gamma \tag{5.22}$$

for γ small. This condition is not usually activated, but serves as an insurance policy against bad CG directions that I investigate in detail later in the section.

ADMM for SVR has been employed using NASA's supercomputer Pleiades to solve aviation safety problems on separate cores and updates a consensus [9]. Their problem is very similar to the time series problem (3.2), but I employ conjugate gradient (CG) for the distributed (5.17) problem whereas they use CVX. A common critique of ADMM is that the algorithm is slow to converge to an accurate solution, but optimising over the inner CG loops gives very fast convergence nonetheless.

5.2 Fast Iterative Shrinkage-Thresholding Algorithm

The set-up for separable non-smooth programmes is similar to that of ADMM, given by

$$\min f(\mathbf{x}) + g(\mathbf{x}) \tag{5.23}$$

where f is differentiable and g is non-smooth but is "reasonable" to work with. Consider a very common linear programme in signal processing, compressed sensing and statistical learning which is the LASSO least-square problem

$$\min \left\{ \left\| \mathbf{A}\mathbf{x} - \mathbf{y} \right\|_2^2 + \lambda \|\mathbf{x}\|_1 \right\} \tag{5.24}$$

where $\mathbf{y}$ is the measured, noisy data (e.g. image, signal, etc.), $\mathbf{A}$ is the measurement operator and $\mathbf{x}$ is the true data. The ℓ_1-norm forces a sparse solution that reduces as much noisy variation as possible in the reconstruction.

The main ingredient of non-smooth programming is the proximal operator, defined as

$$\text{prox}_g(\mathbf{y}) = \underset{x \in X}{\arg\min} \left\{ \frac{1}{2} \left\| \mathbf{x} - \mathbf{y} \right\|_2^2 + \lambda g(\mathbf{x}) \right\} \tag{5.25}$$

where g is a non-smooth function. The proximal operator can be seen as a balanced trade-off between the minimum of the function evaluation and a pre-image distance penaliser from $\mathbf{y}$. If g in (5.22) is "simple", then we have either an analytical form for its proximal operator or a straightforward numerical solution. A closer look at the solution form yields an expression that is suggestive of an updating step. From the first order condition for a stationary point

$$\begin{aligned} 0 &\in \nabla f(\mathbf{x}^*) + \lambda \partial g(\mathbf{x}^*) \\ 0 &\in \tau \nabla f(\mathbf{x}^*) + \tau \partial g(\mathbf{x}^*) - \mathbf{x}^* + \mathbf{x}^* \\ (\mathbf{I}_n + \tau \partial g)(\mathbf{x}^*) &\ni (\mathbf{I}_n - \tau \nabla f)(\mathbf{x}^*) \\ \mathbf{x}^* &= (\mathbf{I}_n + \tau \partial g)^{-1} \circ (\mathbf{I}_n - \tau \nabla f)(\mathbf{x}^*) \end{aligned} \tag{5.26}$$

where ∂ is the subdifferential operator and $(\mathbf{I}_n + \tau \partial g)^{-1}$ is called the resolvent of ∂f. To be more specific, by taking the first-order Taylor approximation of $f(\mathbf{x}^k)$ and inserting this expression into (5.25), we get a forward-backward [6] algorithm updating rule

$$\mathbf{x}^{k+1} = \text{prox}_f \left[\mathbf{x}^k - \tau \nabla f(\mathbf{x}^k) \right] \tag{5.27}$$

where τ is an appropriate step size.

Proposition 3. *The proximal operator for the ℓ_1-norm of a vector $\mathbf{x}$ is given by the soft-thresholding or shrinkage operator applied element-wise*

$$\mathcal{T}_\lambda(\mathbf{x}) = \text{sgn}(\mathbf{x}) \odot (|\mathbf{x}| - \lambda \mathbf{1}_n)^+$$

where $\odot$ is the Hadamard product and $\mathbf{1}_n$ is a vector of ones with the same length as $\mathbf{x}$.

Proof. Using the definition of the proximal operator

$$\begin{aligned} \text{prox}_g(\mathbf{x}) &= \underset{\mathbf{y}}{\arg\min} \left\{ \frac{1}{2} \left\| \mathbf{y} - \mathbf{x} \right\|_2^2 + \lambda \|\mathbf{y}\|_1 \right\} \\ 0 &\in \mathbf{y} - \mathbf{x} + \lambda \partial \|\mathbf{y}\|_1 \end{aligned}$$

and there are two cases for the subdifferential of the ℓ_1-norm

$$\partial|\mathbf{x}_i| = \begin{cases} 1 \times \text{sgn}(\mathbf{x}_i) & \text{if } \mathbf{x}_i \neq 0 \\ \in [-1, 1] & \text{if } \mathbf{x}_i = 0 \end{cases}$$

If $\mathbf{y}_i \neq 0$

$$\text{prox}_g(\mathbf{x}_i) = \begin{cases} \mathbf{y}_i = \mathbf{x}_i - \lambda & \text{if } \mathbf{y}_i > 0 \implies \mathbf{x}_i > \lambda \\ \mathbf{y}_i = \mathbf{x}_i + \lambda & \text{if } \mathbf{y}_i < 0 \implies \mathbf{x}_i < \lambda \end{cases}$$

and if $\mathbf{y}_i = 0$

$$\text{prox}_g(\mathbf{x}_i) = \mathbf{y}_i = 0 \implies \mathbf{x}_i \in [-\lambda, \lambda]$$

Combine into one expression, we get

$$\text{prox}_g(\mathbf{x}_i) = \text{sgn}(\mathbf{x}_i) \times (|\mathbf{x}_i| - \lambda)^+ \ \forall\, i$$

□

From proposition 3, the ISTA algorithm is a simple specialisation of (5.26)

$$\mathbf{x}^{k+1} = \mathcal{T}_\lambda \Big[\mathbf{x}^k - \alpha \nabla f(\mathbf{x}^k) \Big] \tag{5.28}$$

The drawback of this algorithm is that it runs in linear time, i.e. the step convergence is $\mathcal{O}(1/k)$, because it is essentially an altered steepest descent algorithm. Beck and Teboulle [2] introduce FISTA to boost the empirical performance of ISTA with some extra steps that biases every step to produce faster convergence, which is fully described in algorithm 2.

To solve the SVR dual problem with FISTA, I first note that the problem (3.9) described is nearly in FISTA form. Rewriting in matrix notation, the programme reads

$$\begin{aligned} &\min_{\boldsymbol{\alpha}} \Big\{ f(\boldsymbol{\alpha}) + \epsilon g(\boldsymbol{\alpha}) \Big\} \\ &\text{s.t.} \begin{cases} \langle \mathbf{1}_n, \boldsymbol{\alpha} \rangle = 0 \mathbf{1}_n \\ -C\mathbf{1}_n \preccurlyeq \boldsymbol{\alpha} \preccurlyeq C\mathbf{1}_n \end{cases} \end{aligned} \tag{5.29}$$

where $f(\boldsymbol{\alpha}) = (1/2)\boldsymbol{\alpha}^{\mathrm{T}} \mathbf{K} \alpha - \langle \mathbf{Y}, \boldsymbol{\alpha} \rangle$ and $g(\boldsymbol{\alpha}) = \|\boldsymbol{\alpha}\|_1$. The gradient of the smooth part is thus

$$\nabla f(\boldsymbol{\alpha}) = \mathbf{K}\alpha - \mathbf{Y} \tag{5.30}$$

I make two observations: from Dai and Fletcher [8], the box constraint can be satisfied at every iteration through the box projection operator applied element-wise before the proximal projection

$$\tilde{\boldsymbol{\alpha}} = \Pi_C(\boldsymbol{\alpha}) = \text{mid}(-C\mathbf{1}_n, \boldsymbol{\alpha}, C\mathbf{1}_n) \tag{5.31}$$

and that since the equality constraint relates to the bias term b, we can just dispense of the constraint since the optimised result can be appropriately shifted. Specifically, upon retrieval of $\boldsymbol{\alpha}^*$ we can let

$$b = \left\langle \mathbf{Y} - \mathbf{K}\alpha^* \right\rangle \tag{5.32}$$

where $\langle . \rangle$ denotes sample average.

The stopping condition for FISTA depends on the parameter input values, which reads

$$\max \{|\boldsymbol{\alpha}_i^{k+1}|\}_{i=1}^n \leq \gamma \ \bigcap \ \frac{\|\boldsymbol{\alpha}^{k+1} - \boldsymbol{\alpha}^k\|_2}{\|\boldsymbol{\alpha}^k\|_2} \leq tol \tag{5.33}$$

where $\gamma \in \mathbb{R}^+$ repeats its role as an insurance policy like in the ADMM case. The algoritm depends crucially on the threshold set for the soft-tresholding phase, usually set to be a positive number over the Lipschitz constant

$$M = \sup \frac{\|f(y) - f(x)\|_2}{\|y - x\|_2} \ \forall\, y \neq x \tag{5.34}$$

which relates to the degree of non-differentiability of a smooth function.

Proposition 4. *The Lipschitz constant for the gradient of f defined in the FISTA SVR problem (5.28) is given by*

$$M = \|\mathbf{K}\|_F$$

where $\|.\|_F$ denotes the Frobenius norm $\sqrt{\sum_i \sum_j \mathbf{K}_{i,j}^2}$.

Proof. By definition of Lipschitz continuity, $\forall\, \boldsymbol{\alpha}_1 \neq \boldsymbol{\alpha}_2$, $\exists\, M \geq 0$ s.t.

$$\|\nabla f(\boldsymbol{\alpha}_1) - \nabla f(\boldsymbol{\alpha}_2)\|_2 \leq M\|\boldsymbol{\alpha}_1 - \boldsymbol{\alpha}_2\|_2$$

and the gradient of f is given by $\nabla f(\mathbf{x}) = \mathbf{K}\alpha - \mathbf{Y}$, therefore

$$\begin{aligned} \|\nabla f(\boldsymbol{\alpha}_1) - \nabla f(\boldsymbol{\alpha}_2)\|_2 &= \|\mathbf{K}(\boldsymbol{\alpha}_1 - \boldsymbol{\alpha}_2)\|_2 \\ &\leq \|\mathbf{K}\|_2\|\boldsymbol{\alpha}_1 - \boldsymbol{\alpha}_2\|_2 \\ &= \|\mathbf{K}\|_F\|\boldsymbol{\alpha}_1 - \boldsymbol{\alpha}_2\|_2 \end{aligned}$$

□

Algorithm 2: FISTA for SVR dual

All variables $= 0$
$\beta^1 = 1$
Initialise *tol*
while *not converged* **do**
  $\boldsymbol{\alpha}^{k+1} = \mathbf{z}^k - \theta\nabla f(\boldsymbol{\alpha}^k \mid \mathbf{K}, \mathbf{Y})$ | θ chosen from WolfeLineSearch
  $\boldsymbol{\alpha}^{k+1} = \Pi_C(\boldsymbol{\alpha}^{k+1})$
  $\boldsymbol{\alpha}^{k+1} = \mathcal{T}_{\lambda/M}(\boldsymbol{\alpha}^{k+1})$
  $\beta^{k+1} = (1/2)\big(1 + \sqrt{1 + 4(\beta^k)^2}\big)$
  $\mathbf{z}^{k+1} = \boldsymbol{\alpha}^{k+1} + \frac{\beta^k - 1}{\beta^{k+1}}(\boldsymbol{\alpha}^{k+1} - \boldsymbol{\alpha}^k)$
  $\boldsymbol{\alpha}^k = \boldsymbol{\alpha}^{k+1}$
end while
return $\boldsymbol{\alpha}^{k+1}$

Empirically, setting M equal to the largest eigenvalue of $\mathbf{K}$ also does not impede convergence but one must be careful due to rounding errors in the decomposition that may result in complex eigenvalues.

Theorem 5. *Given that*

$$F(\mathbf{x}) = f(\mathbf{x}) + g(\mathbf{x}) \tag{5.35}$$

defined as in 5.22 and $\{\mathbf{x}_k\}$ be generated by FISTA. Then for any $k \geq 1$, $\forall \mathbf{x}^* \in \mathbf{X}_*$

$$\begin{aligned} F(\mathbf{x}^k) - F(\mathbf{x}^*) &\leq \frac{2\Omega M\|\mathbf{x}_0 - \mathbf{x}^*\|_2^2}{(k+1)^2} \\ \frac{\big[F(\mathbf{x}^k) - F(\mathbf{x}^*)\big](k+1)^2}{2M\|\mathbf{x}_0 - \mathbf{x}^*\|_2^2} &\leq \Omega \end{aligned} \tag{5.36}$$

where $\Omega \geq 1$ *relates to the efficiency of line search over a constant step size in the steepest descent step and* M *is the Lipchitz constant of* $\nabla f(\mathbf{x})$.

The convergence theory for FISTA, given by theorem 4 whose proof can be found in [2], proves a performance boost of the sub-linear convergence of ISTA to $\mathcal{O}(1/k^2)$. Of course, since we do not know $\boldsymbol{\alpha}^*$, the numerical results will give a slightly altered version of Ω which makes the replacements $\boldsymbol{\alpha}^* \to \boldsymbol{\alpha}^k$, $\boldsymbol{\alpha}^k \to \boldsymbol{\alpha}^{k+1}$.

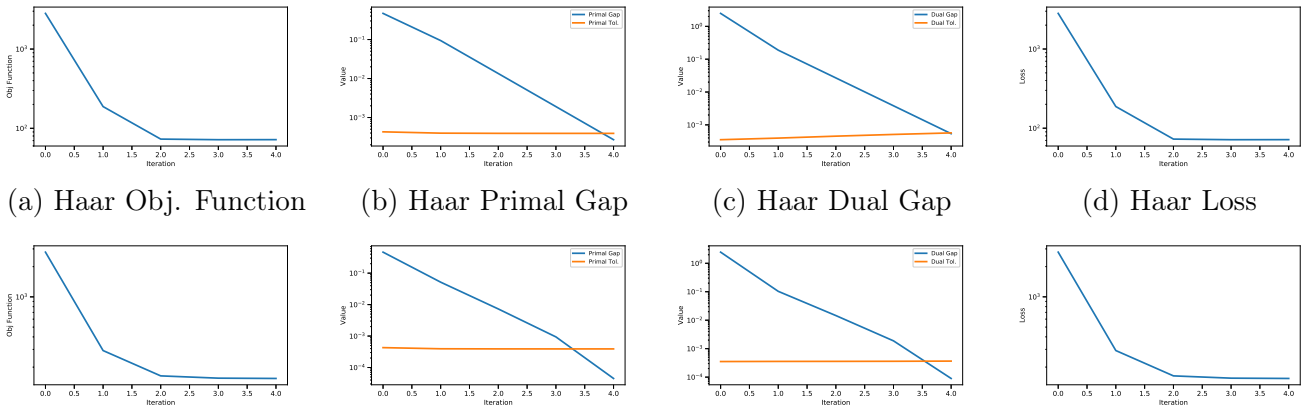
6 Numerical Results

6.1 Baseline Parameters

	Haar						db2					
	1-Lag		4-Lag		20-Lag		1-Lag		4-Lag		20-Lag	
	ADMM	FISTA	ADMM	FISTA	ADMM	FISTA	ADMM	FISTA	ADMM	FISTA	ADMM	FISTA
CPU Time (s)	2.86	8.77	5.02	9.49	5.06	8.35	3.9	8.73	5.18	8.45	5.22	9.5
Iterations	3	1000	5	1000	5	1000	4	1000	5	1000	5	1000
MSE	0.003	0.063	0.013	0.063	0.035	0.066	0.004	0.046	0.016	0.063	0.085	0.130

Table 1: Training Convergence Statistics

I begin with some preliminary information about the baseline parameter values. For ADMM, $C = 1, \epsilon = 0.1, \gamma = 5, \alpha = 1.5, \rho = 1.5$. There is nothing particularly special about these values and an extent, they are arbitrarily chosen from the ranges advised by Boyd et. al [4]. Tolerance for CG is set to $1e^{-3}$ and the maximum number of iterations for ADMM is 1000 and for CG is 10. ADMM tolerances are set per algorithm 1. All CG steps taken via the Polak-Ribière [17] update. The baseline number of subsets for ADMM $p = 4$. For FISTA, $\gamma = 1.5$, $\lambda = 60$, maximum number of iterations is 1000. Step norm convergence tolerance is set at $1e^{-3}$. All codes are written in Python using the PyWavelets package [12]. Due to space constraint, I will primarily focus on the 20-lag prediction in this section because of its high predictive capabilities despite the long delay.



(a) Haar Obj. Function (b) Haar Primal Gap (c) Haar Dual Gap (d) Haar Loss

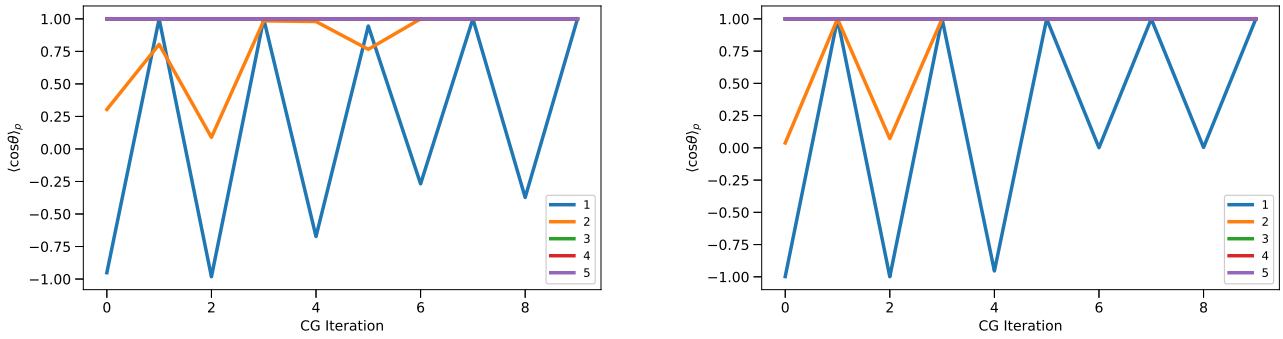
(e) db2 Obj. Function (f) db2 Primal Gap (g) db2 Dual Gap (h) db2 Loss

Figure 4: 20-lag ADMM convergence

ADMM converges in under 6 iterations, but does require the inner CG loops to complete all 10 iterations for each subset, for all outer iterations. The Haar wavelet is almost significantly better at predicting the training output than the Daubechies-2 wavelet under similar convergence time. From figures 4(a,d,e,h), we see that ADMM's convergence is mainly achieved through massive reductions

in the first three iterations, with the loss making up most of the objective function.

This result highly suggests the importance of the approximately solution that the inner CG loops give. Primal and dual gaps close fairly consistently, whereas the objective function reduction and loss function reduction behave much like quadratic convergence even though the algorithm runs in linear time [14]. This is most likely due to the additional penalty reduction that the algorithm has to overcome. Decreasing C to 0.1 pushes the number of iterations needed to converge up to 24 with the final loss and mean squared-error (MSE) virtually the same.



(a) Haar (b) db2

Figure 5: $\langle \cos\theta^k \rangle_p$ over all subsets in ADMM (legend is iteration)

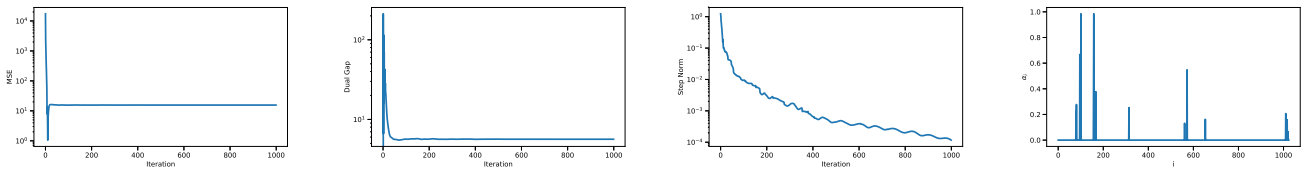
The convergence rate of ADMM depends significantly on the inner-loops' minimisation problems as the outer updates are given analytically. Non-linear CG's convergence theory is at times confounding, given the unstable nature and the fragmentation of empirical observations as noted by [15]. Given a search direction p, an important assumption that convergence theories for nonlinear CG methods make is that

$$\begin{aligned} \sum_{k=1}^{\infty} \cos^2\theta^k \|\nabla f^k\|_2^2 < \infty \\ \cos\theta^k = \frac{\langle -\nabla f(\mathbf{x}^k), p^k \rangle}{\|\nabla f(\mathbf{x}^k)\|_2 \|p^k\|_2} \end{aligned} \tag{6.1}$$

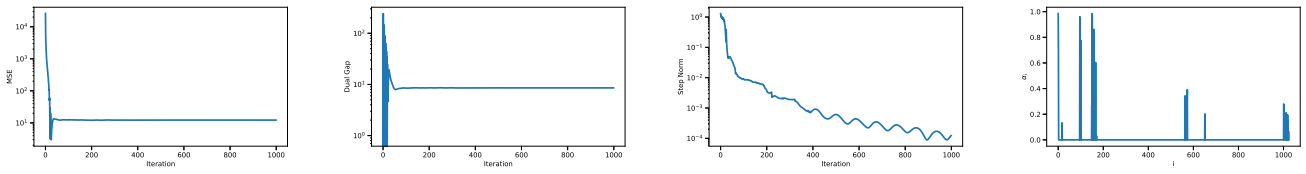
and that $\cos\theta^k$ bounded away from 0. By this criterion, the CG inner loops for ADMM are reasonably robust because the $\cos\theta^k$ mostly bounces back and forth between 1 and -1. From the third iteration onward the directions taken are perfectly correlated with the steepest descent direction.

FISTA convergence is substantially less stable. The algorithm takes all 1000 iterations for both wavelet bases and give similar in-sample accuracy compared to those of ADMM. Sparsity is visualised the convergence plots is defined as $\boldsymbol{\alpha}$ (all elements are non-negative) and I set the zero-measure threshold at < 0.001. The Haar and db2 both achieve approximately 98% sparsity in their weights. The result suggests that only about 2% of variations are necessary to reconstruct the entire yield from the Gaussian kernel. The dual gap for the Haar basis reaches a minimum after fluctuations but that of the db2 basis never seems to regain the minimum value achieved over the span of the iterations.

Figure 7 plots the altered error convergence described in (5.33). The step error plots show that FISTA consistently makes huge jumps toward convergence, even when the size of the steps get smaller. The constant bound, even though not the one proved by Beck and Teboulle, appears to be robust.

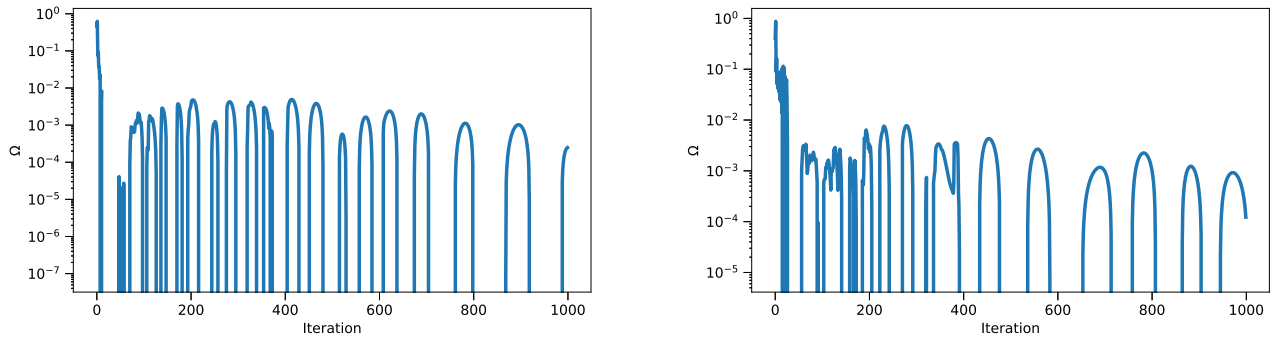


(a) Haar MSE (b) Haar Dual Gap (c) Haar Step Diff. Norm (d) Haar $\boldsymbol{\alpha}$



(e) db2 MSE (f) db2 Dual Gap (g) db2 Step Diff. Norm (h) db2 $\boldsymbol{\alpha}$

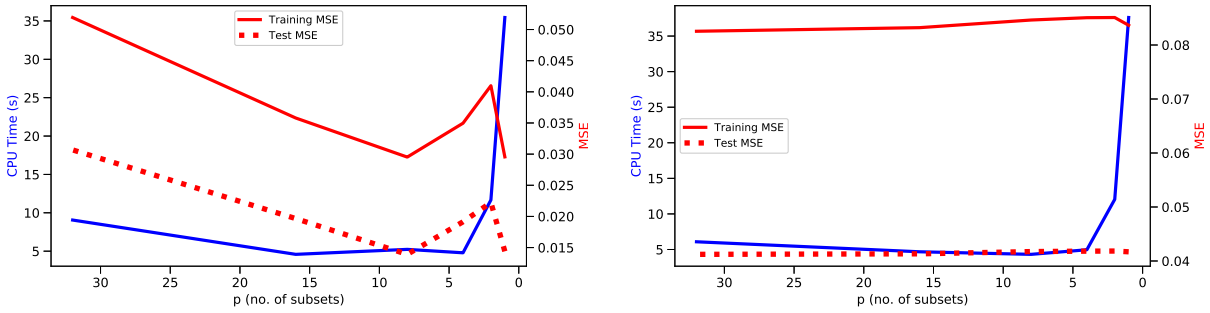
Figure 6: 20-lag FISTA convergence



(a) Haar (b) db2

Figure 7: 20-lag FISTA Step Error Ω^k (5.37)

6.2 Tuning Parameter Values

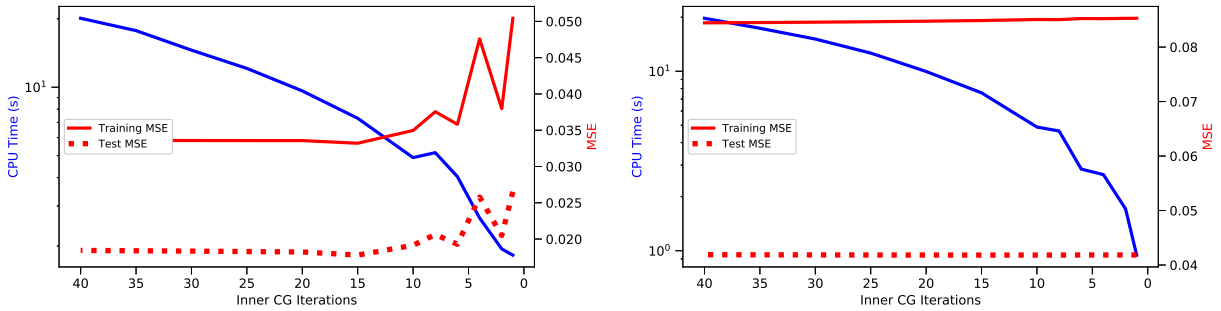


(a) Haar (b) db2

Figure 8: 20-lag ADMM varying number of subets p (holding other baseline param. constant)

Varying the number of ADMM subsets does not yield much improvement in performance, even between the Haar and db2 bases. It seems that the minimum optimisation time given the baseline parameters is achieved at 8 subsets for the Haar wavelet and 4 for the db2 wavelet. ADMM's performance is enhanced considerably in convergence time when the inner CG loop is shortened, as evident in figure 6. The savings can be very significant, reducing from a maximum of around 65 to

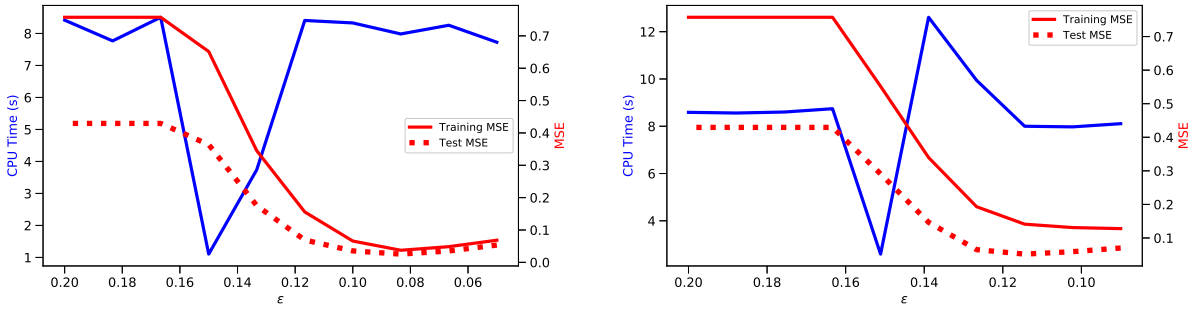
2.5 seconds for the Haar basis and from 47 to 1.2 seconds for the db2 basis. An approximate solution with a good CG direction can push ADMM towards convergence very quickly. The mean squared error is not significantly affected by changing the parameter values. Varying C and ϵ also does not yield substantially different in and out-of-sample accuracy for ADMM.



(a) Haar (b) db2

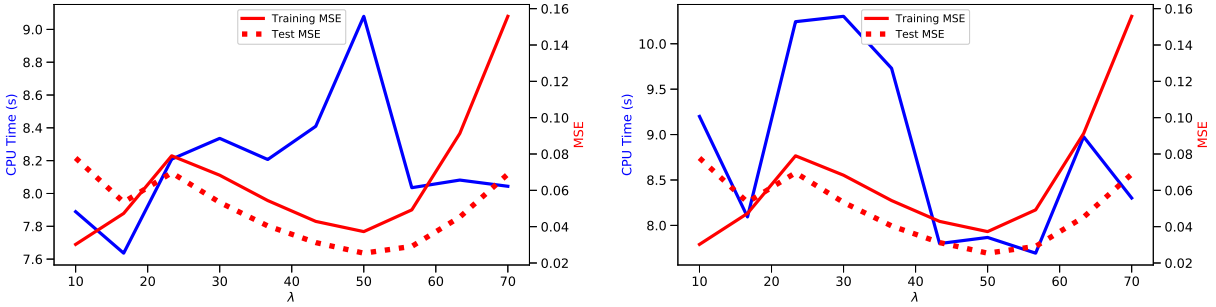
Figure 9: 20-lag ADMM varying inner CG loops' bound (holding other baseline param. constant)

FISTA convergence performance does not change substantially with varying parameter values, but accuracy does change with different values of ϵ and λ. The optimal value for ϵ appears to be around 0.1 for the Haar basis and 0.12 for the db2 basis. The optimal value for λ is 50 for the Haar basis and 60 for the db2 basis. The db2 wavelet is much more sensitive to these values and the search range carried out is significantly smaller than that of the Haar wavelet.



(a) Haar (b) db2

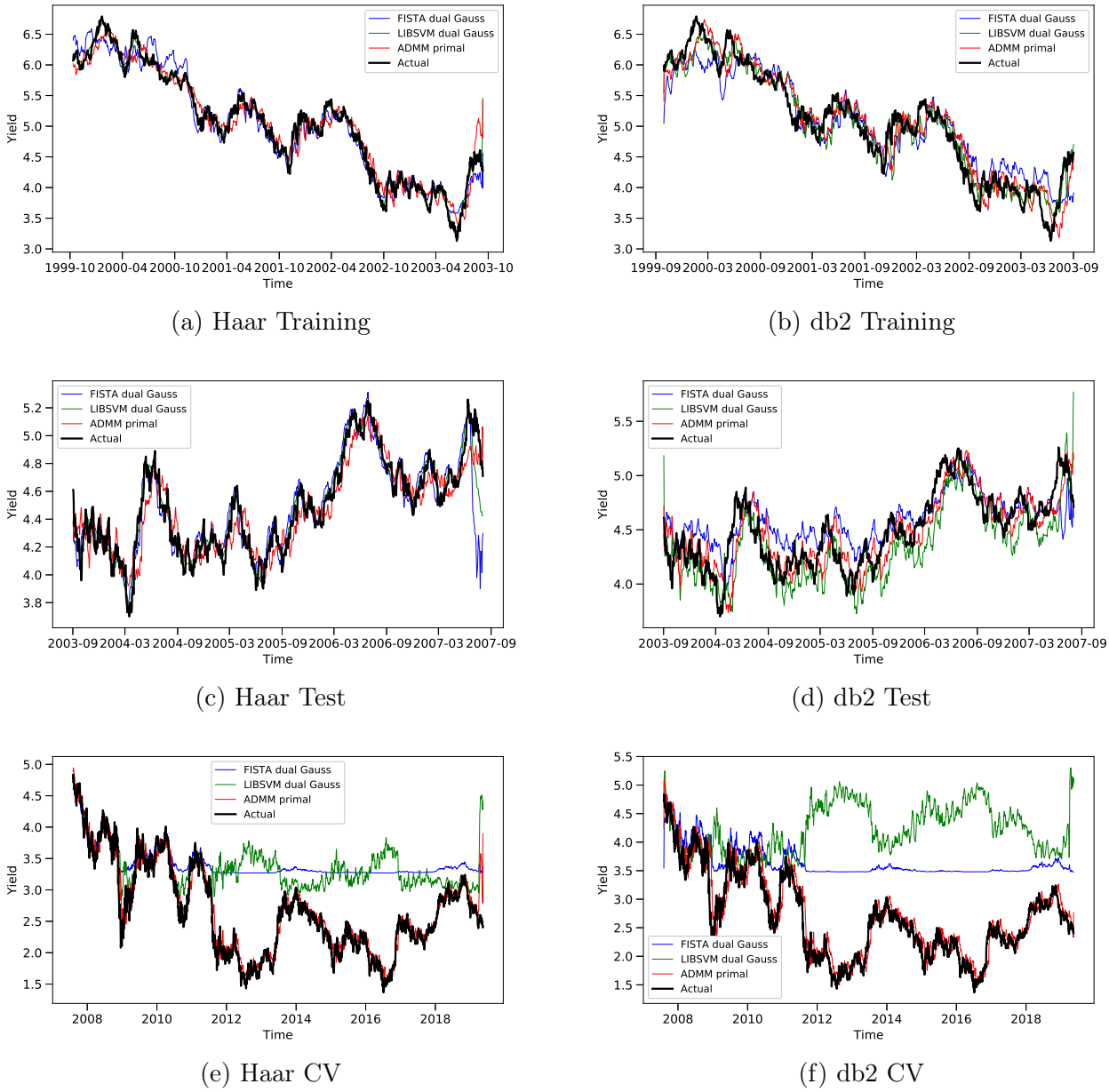
Figure 10: 20-lag FISTA varying ϵ (holding other baseline param. constant)



(a) Haar (b) db2

Figure 11: 20-lag FISTA varying λ (holding other baseline param. constant)

7 Cross Validation



(a) Haar Training

(b) db2 Training

(c) Haar Test

(d) db2 Test

(e) Haar CV

(f) db2 CV

Figure 12: Haar and db2 20-lag prediction vs actual (training and test param. are baseline, CV param. are optimised)

Both wavelets give similar results for the 1-lag and 4-lag prediction of the training set, but at the 20-lag level, the Haar wavelet is superior compared to db2. Mean squared error and Pearson's correlation with the actual yield is given in table 3. The notable statistics are the 20-lag predictions, which have 0.91 to 0.972 Pearson's correlation with the training yield for both bases and prediction functions. In addition to the FISTA and ADMM, I also tested the state-of-the-art LIBSVM algorithm that uses the sequential minimal optimiser (SMO) procedure for the dual problem from the SKLEARN package in Python. ADMM performs much better than either FISTA or SMO out of sample. ADMM still performs slightly better than SMO when using the linear kernel.

The 20-lag wavelet coefficients perform worse when attempting to predict sharper shocks, which is expected. I take the next 1024 observations as the test set for the set of weights from the 20-lag wavelet coefficients. The db2 wavelet, however, misses out on future movements by significant amounts compared to the Haar wavelet. Performance difference is more apparent around sharp jumps or drops in the yield shown in figures 7(c,d).

	Haar			db2		
	ADMM	FISTA	LIBSVM	ADMM	FISTA	LIBSVM
Training	0.035	0.037	0.007	0.085	0.154	0.053
Test	0.019	0.025	0.008	0.042	0.061	0.061
Cross Val.	0.031	0.83	0.90	0.060	1.22	3.52

Table 2: 20-lag MSE

FISTA's prediction output varies much more than that of ADMM in the test set, most likely due to over-fitting of the noisy sharp movements in the training set. Because of duality, these differences are more likely caused by differences obtained from the kernel than by the algorithms themselves. Cross-validation performance for FISTA deteriorates significantly after the height of the financial crisis in 2008/2009 and it never recovers the true range of the yield. ADMM, however, is able to capture almost all of the movements in the yield well into late 2015 using both bases. The parameter tuning to the test set appers to only work on ADMM's prediction function.

The Haar wavelet gives considerably better result both in and out of sample. The weights retrieved from ADMM generalise well into the cross-validation set, whereas for FISTA no amount of tuning can get the parameter values to yield comparable performance out of sample.

8 Conclusion

In this paper, I have demonstrated the power of non-parametric time-series forecasting using scale decomposition through discrete wavelet transform. Even at 20-lags, the predicted values have high precision. The dual problem that is widely attacked in literature can be solved just as efficiently in the primal. The results show that there is a large degree of structural persistence and predictability in the evolution of the risk-free rate over time.

9 Bibliography

[1] K. J. Arrow and G. Debreu. Existence of an equilibrium for a competitive economy. *Econometrica*, 22(3):265–290, 1954.

[2] A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM J. Img. Sci.*, 2(1):183–202, Mar. 2009.

[3] F. Black and M. Scholes. The pricing of options and corporate liabilities. *Journal of Political Economy*, 81(3):637–654, 1973.

[4] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Found. Trends Mach. Learn.*, 3(1):1–122, Jan. 2011.

[5] O. Chapelle. Training a support vector machine in the primal. *Neural Comput.*, 19(5):1155–1178, May 2007.

[6] P. Combettes and V. Wajs. Signal recovery by proximal forward-backward splitting. *Multiscale Modeling & Simulation*, 4(4):1168–1200, 2005.

[7] C. Cortes and V. Vapnik. Support-vector networks. *Mach. Learn.*, 20(3):273–297, Sept. 1995.

[8] Y.-H. Dai and R. Fletcher. New algorithms for singly linearly constrained quadratic programs subject to lower and upper bounds. *Mathematical Programming*, 106(3):403–421, May 2006.

[9] K. Das, K. Bhaduri, L. B. Matthews, and C. N. Oza. Large scale support vector regression for aviation safety. In *2015 IEEE International Conference on Big Data (Big Data)*, pages 999–1006, 2015.

[10] FRED, editor. *10-Year Treasury Constant Maturity Rate [DGS10]*. Board of Governors of the Federal Reserve System (US), April 2019.

[11] C. G. Torrence and G. Compo. A practical guide to wavelet analysis. *Bulletin of the American Meteorological Society*, 79:61–78, 01 1998.

[12] G. R. Lee, R. Gommers, K. Wohlfahrt, F. Wasilewski, A. O'Leary, and H. Nahrstaedt. Pywavelets - wavelet transforms in python, 2006.

[13] J. Mercer. Functions of positive and negative type, and their connection with the theory of integral equations. *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character*, 209:415–446, 1909.

[14] R. Nishihara, L. Lessard, B. Recht, A. Packard, and M. I. Jordan. A general analysis of the convergence of admm. In *Proceedings of the 32Nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, pages 343–352. JMLR.org, 2015.

[15] J. Nocedal and S. Wright. *Numerical Optimization*. Springer Series in Operations Research and Financial Engineering. Springer New York, 2006.

[16] D. Percival and H. Mofjeld. Analysis of subtidal coastal sea level fluctuations using wavelets. *Journal of the American Statistical Association*, 92, 10 1997.

[17] E. Polak and G. Ribiere. Note sur la convergence de méthodes de directions conjuguées. *ESAIM: Mathematical Modelling and Numerical Analysis - Modélisation Mathématique et Analyse Numérique*, Volume 3(R1):pp. 35–43, 1969.

[18] A. J. Smola and B. Schölkopf. A tutorial on support vector regression. *Statistics and Computing*, 14(3):199–222, Aug 2004.