Parallelizing Regularized Least Squares

Manuel A. Rivas

Massachusetts Institute of Technology

May 15, 2008

1. Two weeks ago Prof. Edelman covered Support Vector Machines - classification type problems.
2. Today, I will talk about my project on Regularized Least Squares - Ryan Rifkin a former student of Prof. Edelman is well known for his contribution to RLS.
3. Problem : Suppose we have pairs (x_i, y_i) with $x_i \in \mathbb{R}^d$, and we seek a d-dimensional regression function $f(x)$.
4. Solution : We setup the problem as

$$\min_{f \in \mathcal{H}} \left[\sum_{i=1}^{N} L\left(y_i, f(x_i)\right) + \lambda J(f) \right],$$

where $L(y, f(x))$ is a loss function, $J(f)$ is a penalty functional, $\mathcal{H}$ is a space of functions on which $J(f)$ is defined.

- Linear regression, linear discriminant analysis, logistic regression and separating hyperplanes all rely on a linear model.
- Extremely unlikely that the true function $f(X)$ is actually linear in X.
- In regression problems, $f(X) = \mathbb{E}(Y|X)$ will typically be nonlinear.
- Representing $f(X)$ by a linear model is a convenient and at times appropriate approximation to avoid overfitting.

- Kernel functions commonly used:

$$\begin{aligned} \text{linear} &: k(X_i, X_j) = X_i^t X_j \\ \text{polynomial} &: k(X_i, X_j) = (X_i^t X_j + 1)^d \\ \text{gaussian} &: k(X_i, X_j) = exp\left(-\frac{||X_i - X_j||^2}{\sigma^2}\right) \end{aligned}$$

- Define the Kernel matrix K to satisfy $K_{ij} = k(X_i, X_j)$.
- Given an arbitrary point X_*, $k(X, X_*)$ is a column vector whose ith entry is $k(X_i, X_*)$.

- In this project we use the setup :

$$\min_{f \in \mathcal{H}} \left[\sum_{i=1}^{N} (y_i, f(x_i))^2 + \lambda J(f) \right]$$

- This minimizes the weighted sum of the *total* **square** loss. This loss function is great for regression type problems.
- Note: Support Vector Machines for classification use the following loss function

$$V = |1 - yf(x)|_+$$

or the Heavyside step function

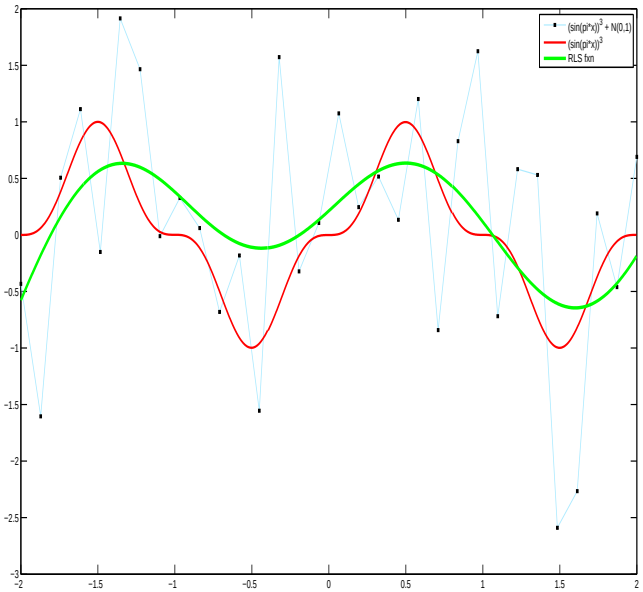
$$V = \theta(-f(x)y).$$

We use Matlab Star-P to parallelize the data. Each processor is assigned a part of the kernel matrix to perform computation. Data Parallelism is exploited.

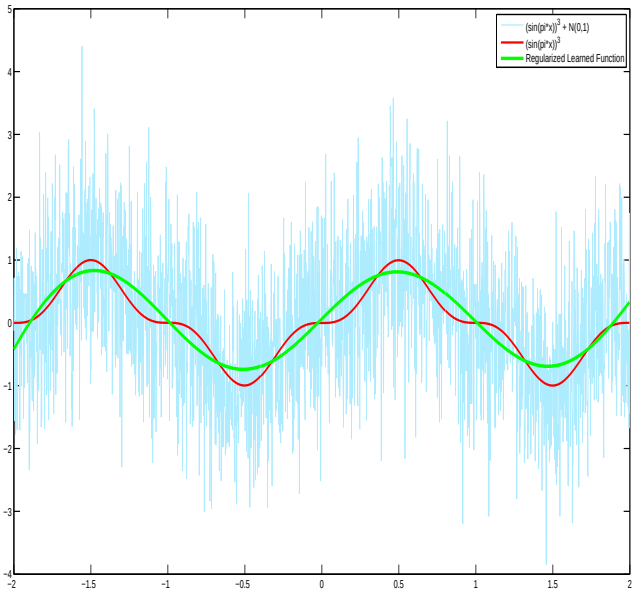
```
xtrain = linspace(-5,5,ntp*p)'; %Uniformly spaced points
 xtest = linspace(-5,5,nptest*p)';%Uniformly spaced points
 ...
 ps = xtrain*diagonalmet*xsup'; %xsup = xtrain; X diag(I) X
 ...
  K = exp(-ps/2);  % Training Kernel
  ...

c = (K+lambda*eye(n))\y;
```

Examples I



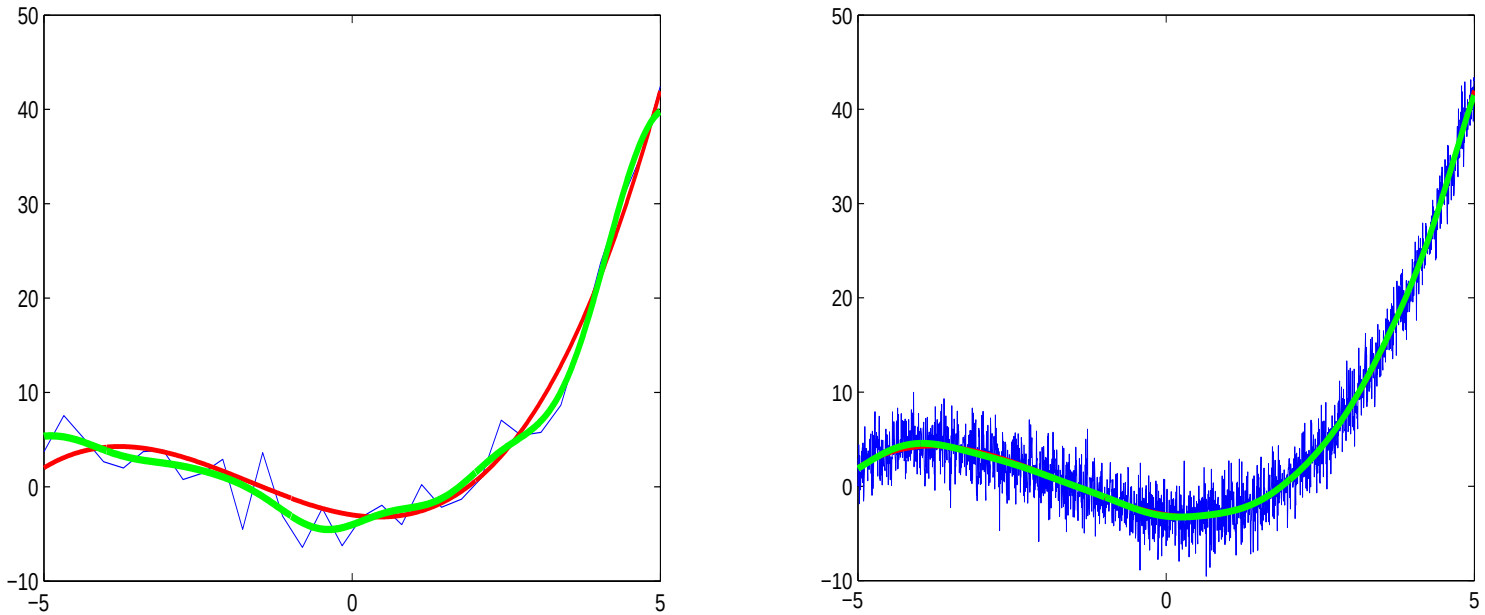
(a) n = 32 training points



(b) n = 2048 training points

Figure: RLS solution for $(\sin(\pi x))^3$

Examples II



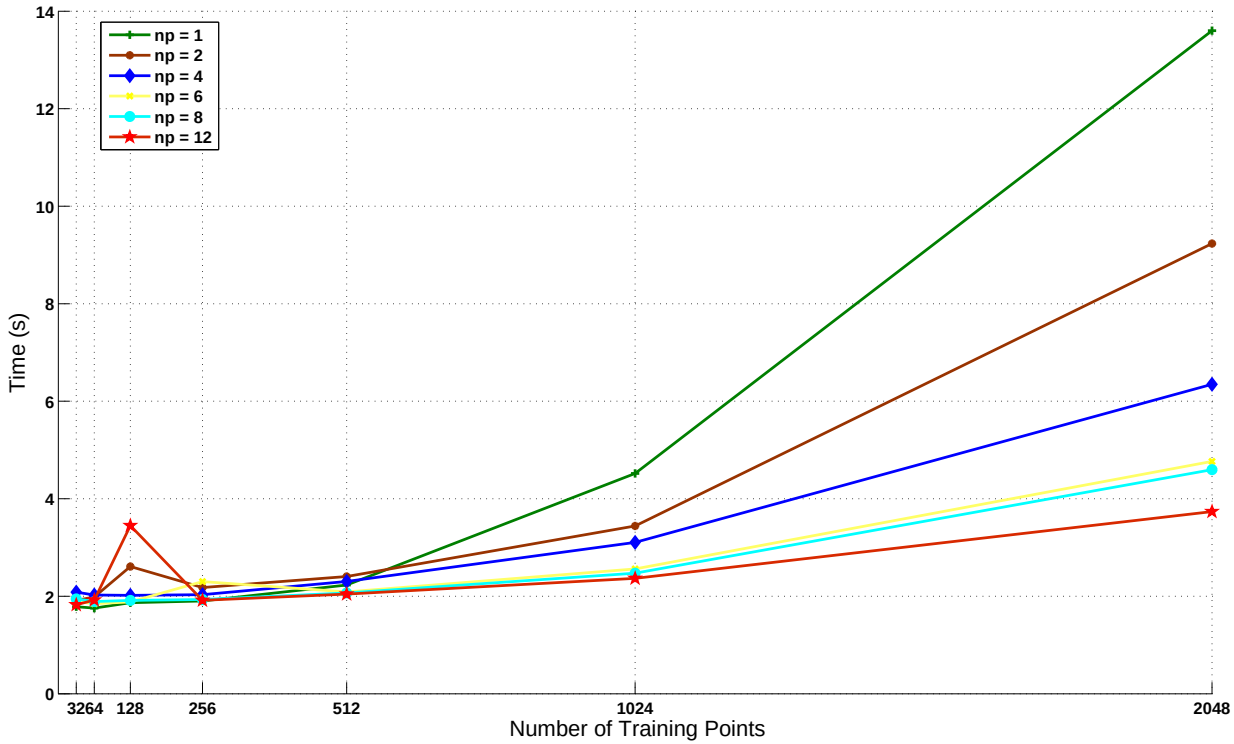
(a) n = 32 training points

(b) n = 2048 training points

Figure: RLS solution for $0.2x^3 + x^2 - x - 3$

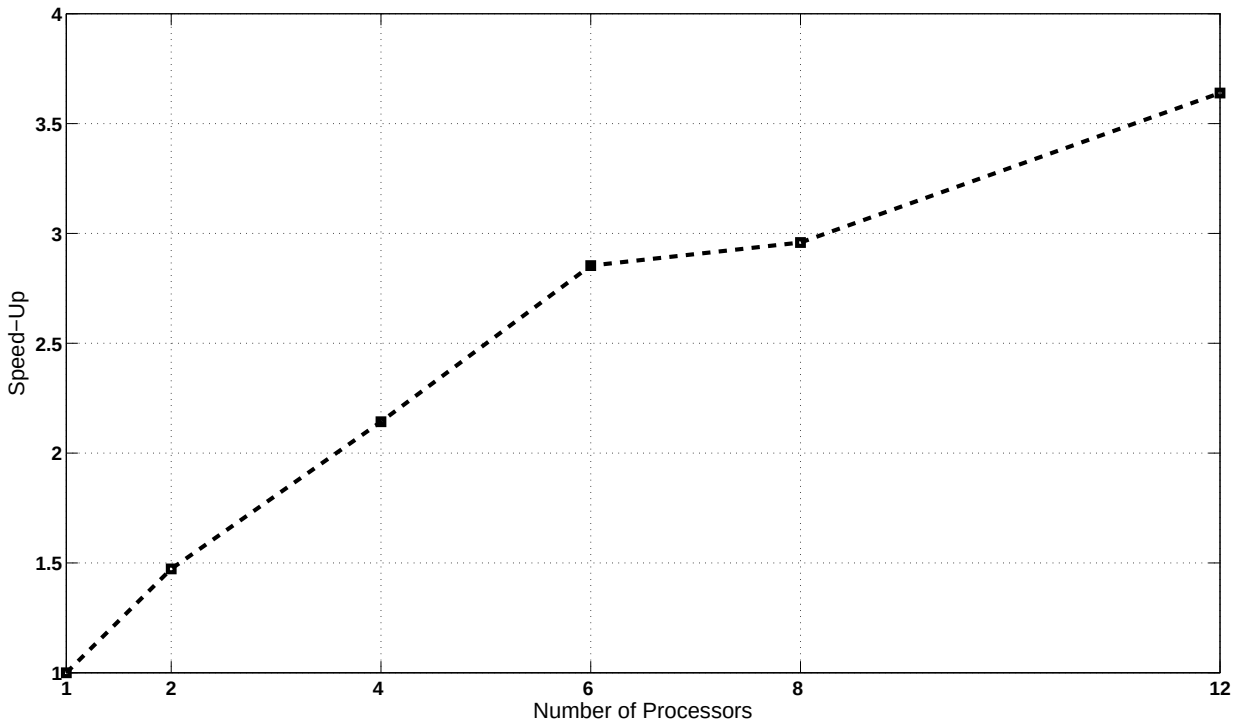
Performance

Figure: Time needed to execute an iteration of RLS with training points = (32,64,128,256,512,1024,2048)



Speedup

Figure: Speedup with number of processors = 1,2,4,6,8,12



Future Work

- As of now the limit on the number of datapoints my implementation handles is ‘ 5,000 datapoints.RLS does not need the QP steps.

Future Work

- As of now the limit on the number of datapoints my implementation handles is ‘ 5,000 datapoints.RLS does not need the QP steps.
- SVM Classification is much more difficult to parallelize - A constrained quadratic programming problem.

Future Work

- As of now the limit on the number of datapoints my implementation handles is ‘ 5,000 datapoints.RLS does not need the QP steps.
- SVM Classification is much more difficult to parallelize - A constrained quadratic programming problem.
- Current literature implementation for SVMs use datasets of 500K to 1M data points requires ‘ 3 days with one machine.

Future Work

- As of now the limit on the number of datapoints my implementation handles is ‘ 5,000 datapoints.RLS does not need the QP steps.
- SVM Classification is much more difficult to parallelize - A constrained quadratic programming problem.
- Current literature implementation for SVMs use datasets of 500K to 1M data points requires ‘ 3 days with one machine.
- Graf H.P., and Vapnik V. (2007) have recently proposed the Spread Kernel Support Vector Machine. Take advantage of the parallelization capability of decomposition algorithms. Idea is to parallelize Kernels and distribute training sets and perform QP independently.