Homework

Stochastic Models and Adaptive Algorithms

Réda Vince

Z697LX

December 31, 2020

Contents

1 Linear regression

1.1 Function approximation with least squares

Ordinary least-squares

At first, a noisy sample of $x \rightarrow y$ is generated. Let $[\mathbf{\Phi}]_{ij} = \mathrm{f}_j(x_i)$ be the transformed input vector and $\mathbf{y}$ the output, where f_j is a basis function. We have to find the $\boldsymbol{\theta}$ parameter vector, for which $\mathbf{\Phi}\,\boldsymbol{\theta} = \mathbf{y}$. From this the $\mathbf{\Phi}(x)$ matrix can be generated after selecting a suitable f. A number of these were tried, and the best one seemed to be the polynomial one, that is $\mathrm{f}_i(x) = x^{i-1}$. As a parameter, $d = 10$ was used. The $\mathbf{\Phi}$ matrix of the sampled inputs and of the LS estimate are the same, so the function is evaluated at the same x values.

For the QR decomposition problem, the "economic" mode of scipy's `qr` function is used, then $\hat{\boldsymbol{\theta}}$ is found with $\hat{\boldsymbol{\theta}} = \mathbf{R}^{-1}\mathbf{Q}^{\mathrm{T}}\mathbf{y}$. Because the pseudoinverse of a matrix is unique, this method gives the same result as the previous one.

The results are shown in figure 1a.

Recursive least-squares

The equation for $\hat{\boldsymbol{\theta}}$ can be reformulated in the following way:

$$\hat{\boldsymbol{\theta}} = \underbrace{\left[\sum_{i=1}^{n} \boldsymbol{\varphi}\,\boldsymbol{\varphi}^{\mathrm{T}}\right]^{-1}}_{\mathbf{\Psi}_n} \underbrace{\sum_{i=1}^{n} y_i\,\boldsymbol{\varphi}_i}_{z_n} . \tag{1}$$

Now we have an update rule for both $\mathbf{\Psi}_{n+1} = \left(\mathbf{\Psi} + \boldsymbol{\varphi}_{n+1}\boldsymbol{\varphi}_{n+1}{}^{\mathrm{T}}\right)^{-1}$, and $z_{n+1} = z_n + \boldsymbol{\varphi}_{n+1}\mathbf{y}_{n+1}$. Both $\mathbf{\Psi}_0$ and z_0 are set to zero. Also, in the code $\boldsymbol{\theta}_n$ is calculated only when plotted for speed.

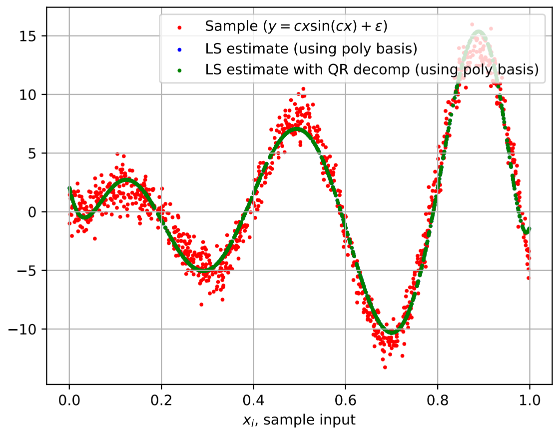
The resulting plots are shown in figure 1c., taken at $n \in [25, 50, 75, 100]$.

Least-norm problem

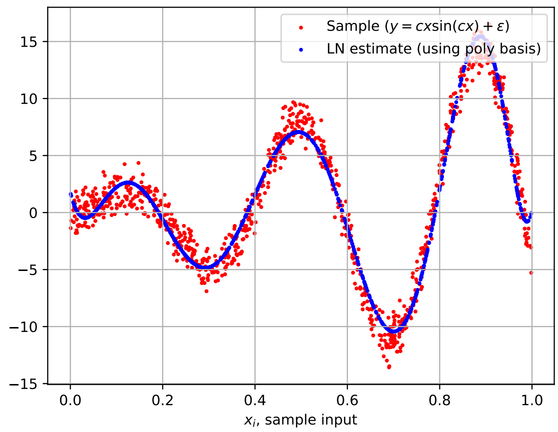
Now let's make $d > n$, which makes $\mathbf{\Phi}$ fat, specifically $n = 100$ and $d = 200$. We make the assumption that $\mathbf{\Phi}$ is still full-rank. The solution is the same, except we are going to use singular value decomposition (SVD) for the pseudoinverse of $\mathbf{\Phi}$.

The SVD is calculated like this: $\mathbf{\Phi}_{d\times n} = \mathbf{U}_{d\times d}\,\mathbf{\Sigma}_{d\times n}\,\mathbf{V}_{n\times n}{}^{\mathrm{T}}$. Let's denote the matrix of column vectors of the normalized eigen-vectors of matrix $\mathbf{\Phi}$ by $\mathrm{eig}(\mathbf{\Phi})$. Let's denote the eigenvalues by $\mathrm{eigval}(\mathbf{\Phi})$. Then, $\mathbf{U} = \mathrm{eig}(\mathbf{\Phi}\,\mathbf{\Phi}^{\mathrm{T}})$, $\mathbf{V} = \mathrm{eig}(\mathbf{\Phi}^{\mathrm{T}}\,\mathbf{\Phi})$ and $\mathbf{\Sigma} = \mathrm{diag}(\mathrm{eigval}(\mathbf{\Phi}\,\mathbf{\Phi}^{\mathrm{T}}))_{d\times n}$. Then, the pseudoinverse is $\mathbf{\Phi}^{+} = \mathbf{V}\,\mathbf{\Sigma}^{+}\,\mathbf{U}^{\mathrm{T}}$. For $\mathbf{\Sigma}^{+}$, we take the inverse of the non-zero elements of $\mathbf{\Sigma}$, and add zeros such that it has the shape of $d \times n$.

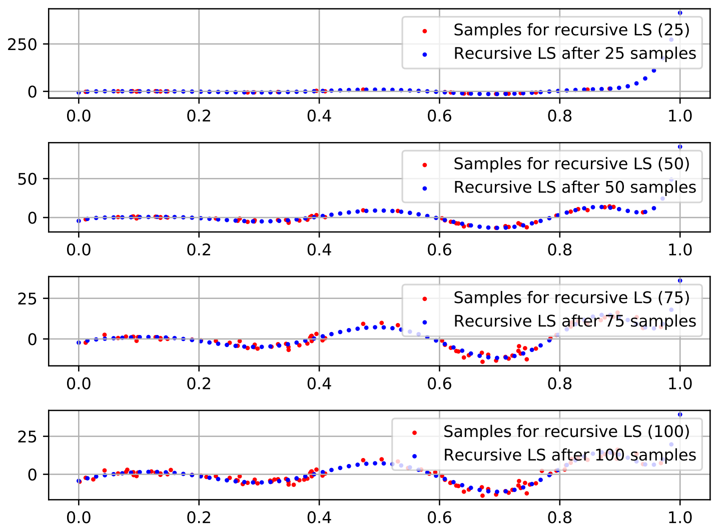
The resulting plots are shown in figure 1b.



(a) Least-squares estimate for thin $\mathbf{\Phi}$



(b) Least-norm estimate for fat $\mathbf{\Phi}$

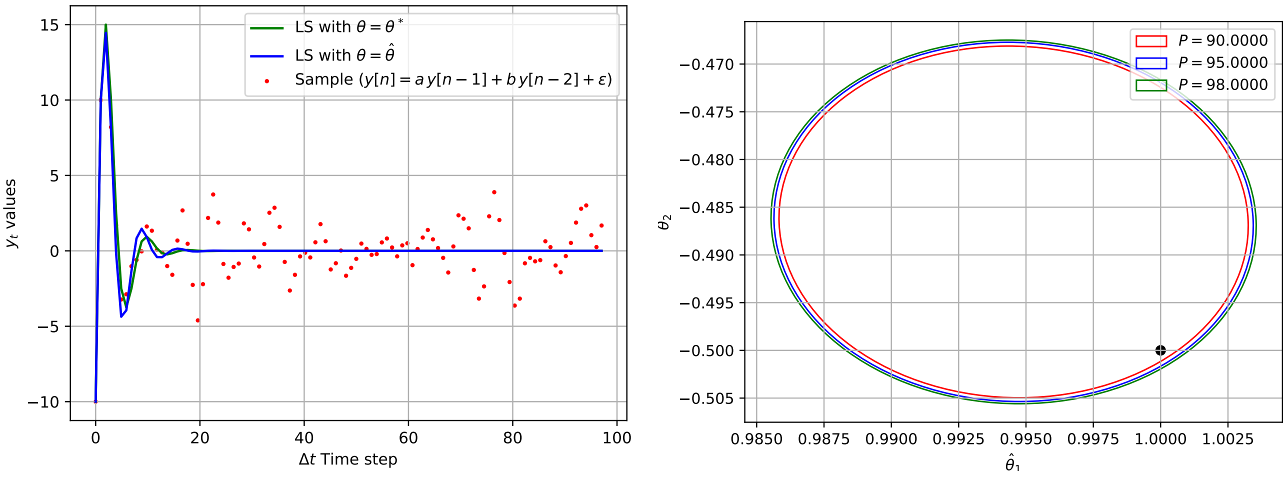


(c) Recursive least-squares

Figure 1: Experiments with ordinary least-squares

1.2 Approximating auto-regressive series

A recursive time-series is generated from the give equation: $y_t = a\, y_{t-1} + b\, y_{t-2} + \epsilon_t$.


LS with $\theta = \theta^*$
LS with $\theta = \hat{\theta}$
Sample $(y[n] = a\,y[n-1] + b\,y[n-2] + \varepsilon)$
y_t values
Δt Time step
$P = 90.0000$
$P = 95.0000$
$P = 98.0000$
θ_2
$\hat{\theta}_1$

(a) (b)

2 Kernel methods

3 Reinforcement learning