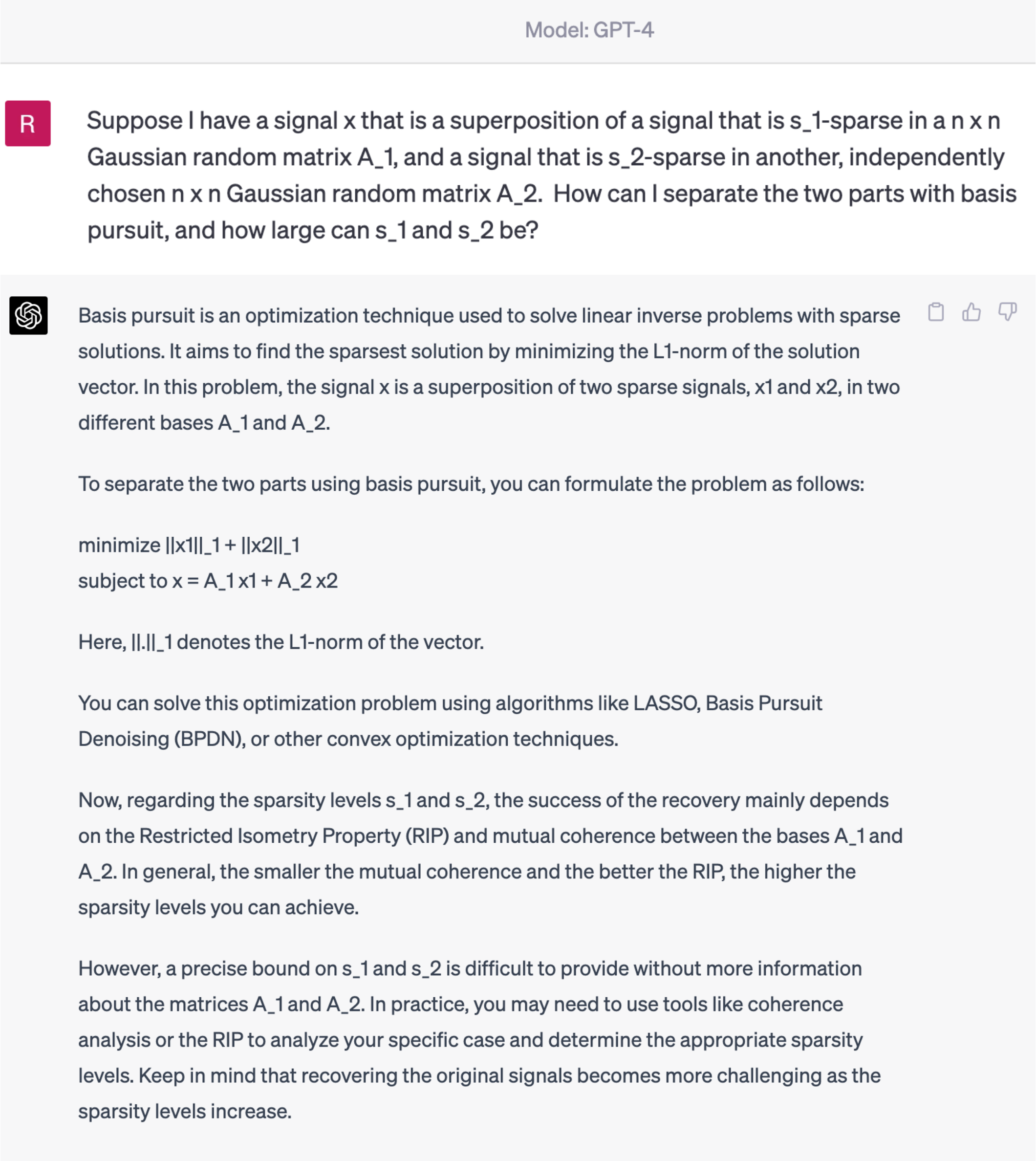
Deep Learning and Inverse Problems
Summer 2024
Lecturer: Reinhard Heckel

Problem Set 2

Issued: Tuesday April 23, 2024, 1:00 pm.
Due: Tuesday April 30, 2024, 1:00 pm.

Problem 1 (ChatGPT for sparse reconstruction)**.** Below is a question and the corresponding response from ChatGPT (generated with the GPT-4 model). Is the algorithm proposed by ChatGPT correct? If yes, can you help to answer how large the sparsity levels s_1 and s_2 can be? Provide references to results from the literature (e.g., from the lecture notes) in your answers.



Model: GPT-4

Suppose I have a signal x that is a superposition of a signal that is s_1-sparse in a n x n Gaussian random matrix A_1, and a signal that is s_2-sparse in another, independently chosen n x n Gaussian random matrix A_2. How can I separate the two parts with basis pursuit, and how large can s_1 and s_2 be?

Basis pursuit is an optimization technique used to solve linear inverse problems with sparse solutions. It aims to find the sparsest solution by minimizing the L1-norm of the solution vector. In this problem, the signal x is a superposition of two sparse signals, x1 and x2, in two different bases A_1 and A_2.

To separate the two parts using basis pursuit, you can formulate the problem as follows:

minimize ||x1||_1 + ||x2||_1
subject to x = A_1 x1 + A_2 x2

Here, ||.||_1 denotes the L1-norm of the vector.

You can solve this optimization problem using algorithms like LASSO, Basis Pursuit Denoising (BPDN), or other convex optimization techniques.

Now, regarding the sparsity levels s_1 and s_2, the success of the recovery mainly depends on the Restricted Isometry Property (RIP) and mutual coherence between the bases A_1 and A_2. In general, the smaller the mutual coherence and the better the RIP, the higher the sparsity levels you can achieve.

However, a precise bound on s_1 and s_2 is difficult to provide without more information about the matrices A_1 and A_2. In practice, you may need to use tools like coherence analysis or the RIP to analyze your specific case and determine the appropriate sparsity levels. Keep in mind that recovering the original signals becomes more challenging as the sparsity levels increase.

1.) Firstly, I would construct the optimization problem as we did in the lecture, for removing the scratches from the image.

→ Define: $x_0 = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$ $\qquad x = A_1 x_1 + A_2 x_2$

Define: $A = \begin{bmatrix} A_1 & A_2 \end{bmatrix}$

→ $\min \|x_0\|_1 \quad \text{s.t.} \quad x = A x_0$

I assume, this doesn't change the optimization problem. However, in terms of s_1 and s_2, despite of what chatgpt said, the A matrix has to have $m \geq 2(s_1+s_2)$ rows.

⟹ That means that we have separate bounds on s_1 and s_2

$$\boxed{\frac{m}{2} \geq s_1 + s_2}$$

In the lecture notes in section 2.3, it is mentioned that A matrix has to have $m \geq 2s$ rows.

Problem 2 (sparse signal recovery)**.** In this problem, we numerically solve a variant of ℓ_1-minimization for sparse reconstruction. Consider the following least-squares regression problem with ℓ_1-norm regularization:

$$\text{minimize}_{\mathbf{x}} g(\mathbf{x}), \quad g(\mathbf{x}) = \frac{1}{2}\|\mathbf{A}\mathbf{x} - \mathbf{y}\|_2^2 + \lambda\|\mathbf{x}\|_1.$$

We take $\mathbf{A} \in \mathbb{R}^{2000\times 1000}$ as a Gaussian random matrix with entries of zero mean and unit variance, and let $\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{e}$, where $\mathbf{e}$ is zero mean Gaussian noise with variance 0.1, and $\mathbf{x}$ is 100-sparse with Gaussian entries, again zero mean and unit variance. Choose λ so that the solution is sparse.

(a) An algorithm to solve this optimization problem numerically is the Iterated Soft Thresholding Algorithm (ISTA), which is an proximal gradient descent algorithm. Define $f(\mathbf{x}) = \frac{1}{2}\|\mathbf{A}\mathbf{x} - \mathbf{y}\|_2^2$, and $\nabla f(\mathbf{x})$ denotes the gradient of f at $\mathbf{x}$. The ISTA algorithm starts at some random initial point $\mathbf{x}^0$ and then performs the iterates:

$$\mathbf{x}^k = \tau_{\lambda\eta}(\mathbf{x}^k - \eta\nabla f(\mathbf{x}^k)).$$

Here, τ is the soft thresholding operator, which is applied entrywise and is defined as

$$\tau_\lambda(x) = \begin{cases} x + \lambda, & \text{if } x < -\lambda, \\ 0, & \text{if } x \in [-\lambda, \lambda], \\ x - \lambda, & \text{if } x > \lambda. \end{cases}$$

Moreover, η is a stepsize parameter. Implement the ISTA algorithm, and provide a plot of convergence of the ISTA algorithm. In this plot, you plot the value of $g(\mathbf{x}^k) - g(\mathbf{x}^*)$ over the number of iterations, k. Here, $\mathbf{x}^*$ is obtained by running the algorithm until convergence.

(b) Next, take $\mathbf{A} \in \mathbb{R}^{500\times 2000}$ as a Gaussian random matrix with zero-mean entries. Generate random k-sparse vectors $\mathbf{x}^* \in \mathbb{R}^{2000}$ (as before, the non-zeros are zero-mean Gaussian) for k ranging from 1 to 500 in some suitably chosen steps, and explore numerically for which values of k, recovery of $\mathbf{x}^*$ is possible from the unterdetermined measurement $\mathbf{y} = \mathbf{A}\mathbf{x}^*$ by solving the least-squares regression problem numerically. Note that you have to tune the parameter λ appropriately. One approach to do so is to perform a grid search of the best parameter λ on a logarithmic scale.

Problem 2 (sparse signal recovery). In this problem, we numerically solve a variant of ℓ_1-minimization for sparse reconstruction. Consider the following least-squares regression problem with ℓ_1-norm regularization:

$$\text{minimize}_{\mathbf{x}} g(\mathbf{x}), \quad g(\mathbf{x}) = \frac{1}{2}\|\mathbf{A}\mathbf{x} - \mathbf{y}\|_2^2 + \lambda\|\mathbf{x}\|_1.$$

We take $\mathbf{A} \in \mathbb{R}^{2000\times1000}$ as a Gaussian random matrix with entries of zero mean and unit variance, and let $\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{e}$, where $\mathbf{e}$ is zero mean Gaussian noise with variance 0.1, and $\mathbf{x}$ is 100-sparse with Gaussian entries, again zero mean and unit variance. Choose λ so that the solution is sparse.

(a) An algorithm to solve this optimization problem numerically is the Iterated Soft Thresholding Algorithm (ISTA), which is an proximal gradient descent algorithm. Define $f(\mathbf{x}) = \frac{1}{2}\|\mathbf{A}\mathbf{x} - \mathbf{y}\|_2^2$, and $\nabla f(\mathbf{x})$ denotes the gradient of f at $\mathbf{x}$. The ISTA algorithm starts at some random initial point $\mathbf{x}^0$ and then performs the iterates:

$$\mathbf{x}^k = \tau_{\lambda\eta}(\mathbf{x}^k - \eta\nabla f(\mathbf{x}^k)).$$

Here, τ is the soft thresholding operator, which is applied entrywise and is defined as

$$\tau_\lambda(x) = \begin{cases} x + \lambda, & \text{if } x < -\lambda, \\ 0, & \text{if } x \in [-\lambda, \lambda], \\ x - \lambda, & \text{if } x > \lambda. \end{cases}$$

Moreover, η is a stepsize parameter. Implement the ISTA algorithm, and provide a plot of convergence of the ISTA algorithm. In this plot, you plot the value of $g(\mathbf{x}^k) - g(\mathbf{x}^*)$ over the number of iterations, k. Here, $\mathbf{x}^*$ is obtained by running the algorithm until convergence.

$f(x) = \frac{1}{2}\|Ax - y\|_2^2$, $Ax - y = g(x)$

$= \frac{1}{2}\|g(x)\|_2^2$

$= \frac{1}{2} g(x)^T g(x)$

$$\frac{\partial f(x)}{\partial x} = g(x)^T \cdot \frac{\partial g(x)}{\partial x} = (Ax - y)^T \cdot A = x^T A^T A - y^T A$$

$$\nabla_x f(x) = \left(\frac{\partial f(x)}{\partial x}\right)^T = A^T A x - A^T y \;//$$

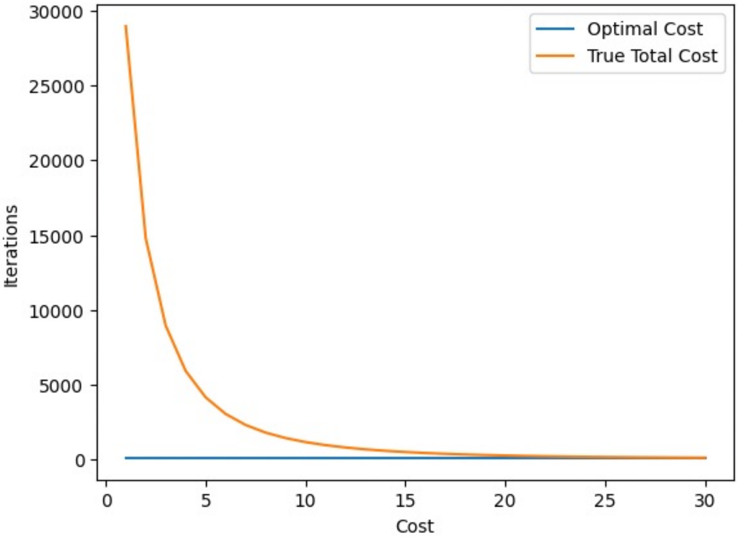
$$x^k = \tau_{\lambda\eta}(x^k - \eta\nabla f(x^k))$$

$$= \begin{cases} x^k - \eta\nabla f(x^k) + \lambda\eta & \text{if } x^k - \eta\nabla f(x^k) < -\lambda\eta \\ 0 & \\ x^k - \eta\nabla f(x^k) - \lambda\eta & \text{if } x^k - \eta\nabla f(x^k) > \lambda\eta \end{cases}$$

choose $\lambda = 0{,}01$

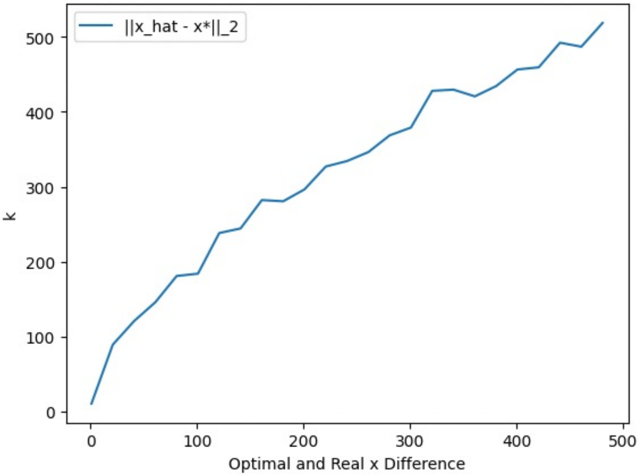
$\eta = \frac{1}{\sigma_{max}^2}$, where σ_{max}^2 is the greatest singular value of A.

Plot 1:



converges until $1e^{-1}$

b)



The plot shows $\|\hat{x} - x^*\|_2$ where with a slight abuse of notation,

$\hat{x}$ is the $\operatorname{argmin}_x \; g(x)$

x^* is the real value in the initialization.

With increasing k, it gets harder to reconstruct x properly.