Homework: Regularization

Clémence Duplat

December 8, 2023

1 Introduction

The concept of ill-posedness was first studied by Jacques Hadamard in the beginning of the 20th century. Hadamard essentially defined a problem to be well-posed ("bien posé") if the solution is unique and if it is a continuous function of the data. Contrary, an ill-posed problem is one that is not uniquely solvable or not continuous as a function of the data i.e. if small perturbation of the data can cause large perturbations in the solution. The typical example of an ill-posed problem is a Fredholm integral equation of the first kind with a square integrable kernel

$$g(s) = \int_a^b K(s,t)f(t)\,dt, \quad c \leq s \leq d$$

with given K and g, where f is an unknown solution.
There are certain finite-dimensional discrete problems that have properties very similar to those of ill-posed problems, such as being highly sensitive to high frequency perturbations. Examples include the discretizations of the above integral equations. We can be more precise and say that a system of the form

$$A\mathbf{x} = \mathbf{b}, A \in \mathbb{R}^{m\times n}$$

is a discrete ill-posed problem if both the following conditions are satisfied:

1. the singular values of A decay gradually to zero
2. the ratio between largest and smallest singular values is large

Criterion 2 implies that the matrix A is ill-conditioned, while criterion 1 implies that there is no "nearby" problem with a well-conditioned coefficient matrix and with well-determined numerical rank.
Being ill-posed is not a fundamental barrier to systems being solvable. Rather, the ill-conditioning means that standard solution procedures are not useful. Indeed, the many small singular values of a discrete ill-posed problem essentially make the problem (numerically) under determined. One has to resort to more sophisticated methods, incorporating additional information about the solution, in order to compute a useful solution. This is the central idea behind regularization methods. When such 'side constraints' are introduced, one must give up the requirement $Ax = \mathbf{b}$ exactly in the linear system and instead seek a solution that provides a fair balance between minimizing some cost function that encodes the side constraints and minimizing the residual norm $\|A\mathbf{x} - \mathbf{b}\|_2$.

1.1 Tikhonov regularization

Undoubtedly, the most common and well-known form of regularization is Tikhonov regularization. Here, the idea is to define the regularized solution $\mathbf{x}_\lambda$ (as a function of λ) as the minimizer of the following weighted combination of the residual norm and a side constraint

$$\mathbf{x}_\lambda = \arg\min_{\mathbf{x}} \{\|A\mathbf{x} - \mathbf{b}\|_2^2 + \lambda^2 \|L(\mathbf{x} - \mathbf{x}^*)\|_2^2\}, \tag{1}$$

where the regularization parameter λ controls the weight given to minimization of the side constraint relative to minimization of the residual norm. The side constraint is captured by the matrix L. One typical example is $L = I_n$. In this case, a large λ (strong regularization) favors a small solution norm at the cost of large residual norm, while a small λ has the opposite effect. Other choices for L are also possible. For example, L can be a discrete version of the second derivative operator. We can even arrange it so that

$$\mathbf{x}_\lambda = \arg\min_{\mathbf{x}} \{\lambda_0^2 \|A\mathbf{x} - \mathbf{b}\|_2^2 + \sum_i \lambda_i^2 \|L_i(\mathbf{x} - \mathbf{x}^*)\|_2^2\}. \tag{2}$$

We are going to assume that $\mathbf{x}^* = 0$. If the different L_i's are (discrete) derivatives, the rightmost term in 2 is called a Sobolev norm.

The Sobolev norm is a way to measure the smoothness of a solution. For constructing the matrix L, defining the discrete derivatives, forward and backward, is crucial. A discrete derivative is an approximation of the derivative at discrete points.
We can start by defining a first order derivative that is given by the difference $x_{i+1} - x_i$ (forward) or $x_i - x_{i-1}$(backward). The matrix L will then look as follow:

$$L_{\text{forward}} = \begin{bmatrix} -1 & 1 & 0 & \cdots & 0 \\ 0 & -1 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & -1 & 1 \end{bmatrix}$$

$$L_{\text{backward}} = \begin{bmatrix} 1 & -1 & 0 & \cdots & 0 \\ 0 & 1 & -1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 & -1 \end{bmatrix}$$

A second order discrete derivative approximates the rate change of the first derivative and we can use central derivatives. The derivative can be defined by $\frac{x_{i+2}-2x_{i+1}+x_i}{h^2}$ (forward) and $\frac{x_i-2x_{i-1}+x_{i-2}}{h^2}$ (backward), where h is the step size between two discrete points. Here is an example:

$$L = \begin{bmatrix} 1 & -2 & 1 & 0 & \cdots & 0 \\ 0 & 1 & -2 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 & -2 & 1 \end{bmatrix}$$

These are not the only possible such L_i's. Another important class is seminorm matrices, which include norm constraints only on a subdomain of the solution and boundary condition restraints.
"A seminorm matrix, is matrix whose norm is the seminorm, a vector space norm that need to be postive definite." (Definition from Wikipedia)
First, thanks to this matrix, we will be able to focus on specific parts of the solution while leaving others unaffected. If we want to regularize the first k elements of x(we suppose that the size of x is n), our seminorm matrix L could be a matrix containing the identity matrix with size $n \times k$ and zeros everywhere from k to n. This way we will impose constraints only on the k first elements of x.

Then, we can enforce the boundary conditions by penalyzing deviations from the initial value boundary conditions. To make it a simple example, we can define the boundary conditions as follows: $x_0 = 0$ and$x_n = 0$. The matrix L will then have zero values everywhere except on the boundary elements. This way, the other values will not be affeted by this matrix L.

To show that the expression 2 is equivalent to equation 1 with L the Cholesky factor of $\sum_i \lambda_i^2 L_i^T L_i$, we will need to show that

$$\sum_i \lambda_i^2 \|L_i(\mathbf{x} - \mathbf{x}^*)\|_2^2 \ = \lambda^2 \|L(\mathbf{x} - \mathbf{x}^*)\|_2^2$$

We supposed that $\lambda_0 = 1$ in order to focus only into the balance between the different regularization terms and not on the control of the regularized solution. If it was not the case, the Cholesky factor would be $\lambda_0^2 * I_n + \sum_i \lambda_i^2 L_i^T L_i$.
By the Cholesky factor, we can expand the first term as follow:

$$\sum_i \lambda_i^2 (\mathbf{x} - \mathbf{x}^*)^T L_i^T L_i (\mathbf{x} - \mathbf{x}^*)$$

We can then defining the Cholesky decomposition of a positive definite matrix A as follows: $A = LL^T$, with L a lower triangular matrix. For clarity, we will use R to avoid any confusion with my seminorm matrix L and thus we have:

$$\sum_i \lambda_i^2 L_i^T L_i = RR^T$$

If we replace that in our equation we will have :

$$(\mathbf{x} - \mathbf{x}^*)^T RR^T (\mathbf{x} - \mathbf{x}^*) = \lambda^2 \|L(\mathbf{x} - \mathbf{x}^*)\|_2^2$$

by setting $L = R^T$ *CQFD*

This way, we showed that we could simplify the problem into a single matrix norm form. This can be done if L is positive definite, which is the case with seminorm matrices.

With λ one also controls the sensitivity of the regularized solution $\mathbf{x}_\lambda$ to perturbations in A and $\mathbf{b}$, and the perturbation bound is proportional to λ^{-1}.

Numerical methods for actually computing an optimal λ will be discussed later. You have seen that in case of $L = I_n$, the solution is given by

$$\begin{aligned}\mathbf{x}_\lambda &= \sum_{i=1}^{n} \frac{\sigma_i^2}{\sigma_i^2 + \lambda^2} \frac{\mathbf{u}_i^T \mathbf{b}}{\sigma_i} \mathbf{v}_i \\ &= \sum_{i=1}^{n} f_i \frac{\mathbf{u}_i^T \mathbf{b}}{\sigma_i} \mathbf{v}_i\end{aligned}$$

We can define the following theorem:

Theorem 1. *Given matrices $A \in \mathbb{R}^{m\times n}$ and $B \in \mathbb{R}^{p\times n}$ with $m \geq n \geq p$, there exists orthogonal $U \in \mathbb{R}^{m\times n}$ and $V \in \mathbb{R}^{p\times p}$ together with an invertible $X \in \mathbb{R}^{n\times n}$ such that*

$$A = U \begin{pmatrix} \Sigma & \\ & I_{n-p} \end{pmatrix} X^{-1}$$

and

$$B = V[M \;\; 0]X^{-1}$$

with both $\Sigma = diag(\sigma_1, \ldots, \sigma_p)$ and $M = diag(\mu_1, \ldots, \mu_p)$ diagonal matrices with nonnegative entries such that $0 \leq \sigma_1 \ldots \leq \sigma_p \leq 1 \geq \mu_1 \geq \ldots \geq \mu_p > 0$. The values $\gamma_i := \sigma_i/\mu_i$ are referred to as the generalized singular values. In addition we have $\mu_i^2 + \sigma_i^2 = 1$.

If L is not the identity we have the following generalized problem:

$$\mathbf{x}_\lambda = \arg\min_{\mathbf{x}} \{\|A\mathbf{x} - \mathbf{b}\|_2^2 + \lambda^2 \|L(\mathbf{x})\|_2^2\}, \tag{3}$$

We can use the theorem 1 by defining A=A and B=L, then we get:

$$\mathbf{x}_\lambda = \arg\min_{\mathbf{x}} \{\|U \begin{pmatrix} \Sigma & \\ & I_{n-p} \end{pmatrix} X^{-1}\mathbf{x} - \mathbf{b}\|_2^2 + \lambda^2 \|V[M \;\; 0]X^{-1}(\mathbf{x})\|_2^2\}, \tag{4}$$

We can substitute $y = X^{-1}x$ and use the orthogonality of U and V so that we get :

$$\mathbf{x}_\lambda = \arg\min_{\mathbf{y}} \{\| \begin{pmatrix} \Sigma & \\ & I_{n-p} \end{pmatrix} \mathbf{y} - U^T\mathbf{b}\|_2^2 + \lambda^2 \|[M \;\; 0]\mathbf{y}\|_2^2\}, \tag{5}$$

To get rid of the matrix we can decompose y in two terms such that y_1 correspond to the size of Σ and y_2 to the size of I_{n-p}. This way we can also

decompose $U^T b$ in two terms: b_1 that corresponds to the vector alligned with non-zero singular values in Σ and b_2 that corresponds to the vector alligned with the identity I_{n-p}. Therefore, we will have:

$$\mathbf{x}_\lambda = \arg\min_{\mathbf{y}} \{\|\Sigma \mathbf{y_1} - \mathbf{b_1}\|_2^2 + \|\mathbf{y_2} - \mathbf{b_2}\|_2^2 + \lambda^2 \|M \mathbf{y_1}\|_2^2\}, \tag{6}$$

We can use the new filter factors (because the ordering of singular values has changed)

$$f_i = \frac{\gamma_i^2}{\gamma_i^2 + \lambda^2}$$

with $\gamma_i := \sigma_i / \mu_i$ in order to get $y_1 = \sum_{i=1}^{p} f_i \frac{\mathbf{u}_i^T \mathbf{b}}{\sigma_i}$ (because we see it correspond to L=I) and $y_2 = b_2 = \sum_{i=p+1}^{n} \mathbf{u}_i^T \mathbf{b}$. (for this one we don't need the filters factors).
The goal is to find x so will transform y back to x as follow: $x = Xy = x_i b_1 + x_i b_2$. Because we know that $b_2 = U_{n-p}^T b$ we obtain the final expression:

$$\mathbf{x}_\lambda = \sum_{i=1}^{p} f_i \frac{\mathbf{u}_i^T \mathbf{b}}{\sigma_i} \mathbf{x}_i + \sum_{i=p+1}^{n} \mathbf{u}_i^T \mathbf{b}\, \mathbf{x}_i$$

We saw that the new ordering of σ is significant because it changes how we interpret the components of the GSVD in terms of their contributions to the overall structure of the matrices A and B.

Similarly to the usual discrete Picard condition, the more general case for L also leads to a Picard condition, but now in terms of the generalized singular values. That is, the discrete Picard condition in case of $L \neq I_n$ is satisfied whenever the $|u_i^T b|$ (with U from the GSVD) decay at least as fast as the generalized singular values.

1.2 TSVD/TGSVD and DSVD/DGSVD regularization

A fundamental observation regarding the Tikhonov method is that it circumvents the ill-conditioning of A by introducing a new problem with a well-conditioned coefficient matrix with full rank. This matrix is defined by the augmented matrix A with λL : $\begin{pmatrix} A \\ \lambda L \end{pmatrix}$. This way, the regularized problem is defined as follow:

$$\arg\min\{\| \begin{pmatrix} A \\ \lambda L \end{pmatrix} x - \begin{pmatrix} b \\ \lambda L x^* \end{pmatrix} \|_2\}$$

A different way to treat the ill-conditioning of A is to **derive a new problem with a well-conditioned rank deficient coefficient matrix** (well-conditioned meaning that the nonzero singular values span a limited range). A fundamental result about rank deficient matrices is the Eckart-Young theorem. It states that the closest rank-k approximation A_k to A (measured in 2-norm) is obtained by truncating the SVD expansion at k, i.e., A_k is given by

$$A_k = \sum_{i=1}^{k} \sigma_i \mathbf{u}_i \mathbf{v}_i^T$$

The truncated SVD (TSVD) regularization method is based on this observation in that one solves the problem

$$\min \|\mathbf{x}\|_2 \quad s.t. \quad \min \|A_k \mathbf{x} - \mathbf{b}\|.$$

The solution to this problem is given by

$$\mathbf{x}_k = \sum_{i=1}^{k} \frac{\mathbf{u}_i^T \mathbf{b}}{\sigma_i} \mathbf{v}_i$$

Let us note that the TSVD solution $\mathbf{x}_k$ is the only solution that has no component in the numerical null-space of A, spanned by columns of V with numbers from $k+1$ to n. Instead of using filter factors 0 and 1 as in TSVD, one can introduce a smoother cut-off by means of filter factors f_i defined as

$$f_i = \frac{\sigma_i}{\sigma_i + \lambda}$$

thus getting the damped SVD (DSVD). The new filter factors decay slower than Tikhonov filter factors and thus introduce less filtering. We can extend these to the generalized context as well. In this case we have for the truncated GSVD (TGSVD) that

$$\mathbf{x}_k = X \begin{pmatrix} \hat{\Sigma}_k & \\ & I_{n-p} \end{pmatrix} U^T \mathbf{b}$$

in which $\hat{\Sigma}_k = \text{diag}(0, \ldots, 0, \sigma_{p-k+1}^{-1}, \ldots, \sigma_p^{-1})$.
If $L = I$, we know that the regularization matrix L does not introduce any additional structure or constraints into the problem and thus the singular

values of A are not changed (note that in SVD singular values are in a decreasing order in Σ and it is the opposite for TSVD) . In other words the GSVD of A and L will reduce to the SVD of A, because L is the identity matrix and thus $p = n$. We find back:

$$\mathbf{x}_k = X \begin{pmatrix} 0 & & & \\ & \ddots & & \\ & & \sigma_{n-k+1}^{-1} & \\ & & & \ddots \\ & & & & \sigma_n^{-1} \end{pmatrix} U^T \mathbf{b}$$

We see that TGSVD is the same as the TSVD if we consider U and X as the left and right singular vectors of A. It is easy to understand that because the singular values has a different order, the columns of the singular vectors are not in the same order either.

To show this numerically, we will first show that the singular values, when putting in the same order, are the same for TSVD and TGSVD. It is the case. Then I tried to compute the solutions using above formulas but I failed to implement it, I think it is maybe because I due to the fact that I do not order the singular vectors in the right way.

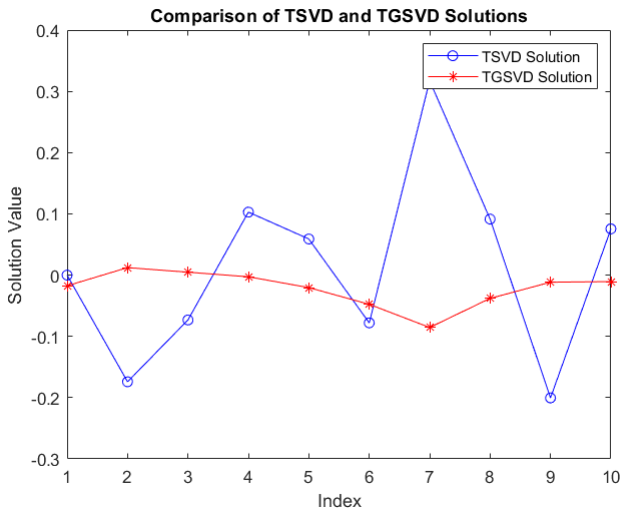

Figure 1: Comparison 2 solutions

Again, we can dampen this by, rather than using filter factors 0 and 1, using the filter factors

$$f_i = \frac{\gamma_i}{\gamma_i + \lambda}.$$

1.3 Conjugate gradient regularization

The conjugate gradient method is an iterative Krylov method for the solution of linear systems. It is only defined for symmetric positive semi-definite systems, but it can in our case be applied to the normal equations

$$A^T A\mathbf{x} = A^T\mathbf{b}$$

the solution of which will be called $\mathbf{x}_*$. We will assume that $A^T A$ is positive definite. This is given in algorithm 1.

Algorithm 1: Conjugate Gradient Algorithm for the normal equations (CGLS)

input : Starting vector $\mathbf{x}_0$, number of iterations k
output: CG solution $\mathbf{x}_k$

init *compute initial residual* $\mathbf{r}_0 := \mathbf{b} - A\mathbf{x}_0$ *and initial auxiliary vector* $\mathbf{d}_0 := A^T\mathbf{r}_0$
for $i = 1 \dots k$ **do**
  $a_i = \|A^T\mathbf{r}_{i-1}\|_2^2 / \|\mathbf{d}_{i-1}\|_A^2$
  $\mathbf{x}_i = \mathbf{x}_{i-1} + a_i\mathbf{d}_{i-1}$
  $\mathbf{r}_i = \mathbf{r}_{i-1} - a_i A\mathbf{d}_{i-1}$
  $\beta_i = \|A^T\mathbf{r}_i\|_2^2 / \|A^T\mathbf{r}_{i-1}\|_2^2$
  $\mathbf{d}_i = A^T\mathbf{r}_i + \beta_i\mathbf{d}_{i-1}$
end

Typically, $\mathbf{x}_0 = 0$ is assumed as the initial guess. This algorithm will, given a starting vector x_0 and a number of iterations k, find a Conjugate Gradiant (CG) solution x_k. The algorithm works as follow :

- initialize a residual $\mathbf{r}_0 := \mathbf{b} - A\mathbf{x}_0$ and an auxilary vector $\mathbf{d}_0 := A^T\mathbf{r}_0$. The residual is measure of error in the current solution and the auxiliary vector is the search direction for the next solution. We can see that it depends on the residual, to be sure it will converge.

- iterate over i and compute the step size a_i based on the residual and the auxiliary vector + update residual, solution and auxiliary vector. The step size is there to ensure convergence by making a balance between fast convergence ans stability. We update x in order to have less error and we update the residual to see how the change in the solution is. We update the auxiliary vector such that it is connected to the previous directions, also to ensure convergence.

We can show that $\mathbf{r}_i = \mathbf{b} - A\mathbf{x}_i$.
In the algorithm they say that the residual is updates as follow:

$$\mathbf{r}_i = \mathbf{r}_{i-1} - a_i A\mathbf{d}_{i-1} \tag{7}$$

Because we know that $\mathbf{r}_0 := \mathbf{b} - A\mathbf{x}_0$, it should be logic to think that $\mathbf{r}_{i-1} := \mathbf{b} - A\mathbf{x}_{i-1}$ will be true for each i smaller than k. We want to be sure it is also true for each i. If we substitute this assumption in equation 7, we will get:

$$\mathbf{r}_i := \mathbf{b} - A\mathbf{x}_i - a_i A\mathbf{d}_{i-1}$$

$$\mathbf{r}_i := \mathbf{b} - A(\mathbf{x}_{i-1} + a_i\mathbf{d}_{i-1})$$

We also know that the solution is updated as $\mathbf{x}_i = \mathbf{x}_{i-1} + a_i\mathbf{d}_{i-1}$ and therefore we will find:

$$\mathbf{r}_i := \mathbf{b} - A\mathbf{x}_i$$

We can introduce some notations:

- $\mathbf{r}_i^{LS} = A^T\mathbf{r}_i$
- $\mathcal{K}_i(A^TA, \mathbf{r}_0^{LS}) = \text{span}\{\mathbf{r}_0^{LS}, A^TA\mathbf{r}_0^{LS}, \ldots, (A^TA)^{i-1}\mathbf{r}_0^{LS}\}$ is the Krylov subspace associated to the CGLS iteration i.
- $\mathbf{e}_i := \mathbf{x}_* - \mathbf{x}_i$
- $\langle \mathbf{v}, \mathbf{w} \rangle_A := \langle A\mathbf{v}, A\mathbf{w} \rangle$, and the norm $\|\mathbf{v}\|_A$ and orthogonality $\mathbf{v} \perp_A \mathbf{w}$ are defined correspondingly.

Now the conjugate gradient iteration can be described completely as the system of recurrences that generates the (unique) sequence of iterates satisfying $\{\mathbf{x}_i \in \mathbf{x}_0 + \mathcal{K}_i(A^TA, \mathbf{r}_0^{LS})\}_i$ such that at step i, $\|\mathbf{e}_i\|_A$ is minimized.

Firstly, we will show that $\|\mathbf{e}_i\|_A$ being minimized in this case is equivalent to $\mathbf{e}_i \perp_A \mathcal{K}_i(A^TA, \mathbf{r}_0^{LS})$. In other words we want to prove that the error at iteration i is orthogonal to the Krylov subspace spanned by span$\{\mathbf{r}_0^{LS}, A^TA\mathbf{r}_0^{LS}, \ldots, (A^TA)^{i-1}\mathbf{r}_0^{LS}\}$ with respect to the inner product defined by A. We can define $\|\mathbf{e}_i\|_A = \|A\mathbf{e}_i\|_2 = \sqrt{\mathbf{e}_i^TA^TA\mathbf{e}_i}$. If we take the definition of orthogonality given we $\langle A\mathbf{v}, A\mathbf{e}_i\rangle = 0$ for every v in the Krylov space. If we deviate this orthogonality it would result in a larger A-norm of the error and it would not minimize it anymore. *CQFD*

Then, we can show that $\mathbf{r}_i^{LS} = A^*A\mathbf{e}_i$. Indeed, if we take the notations above we have $\mathbf{r}_i^{LS} = A^*\mathbf{r}_i$, we know the expression of $r_i = b - A(x_* - e_i)$ and we know that conjuagte gradient regulariation can be applied to the normal equations $A^*Ax_* = A^*b$. Thus $\mathbf{r}_i^{LS} = A^*(\mathbf{b} - A\mathbf{x}_*) = A^*\mathbf{b} - A^*A\mathbf{x}_i + A^*Ae_i = A^*A\mathbf{x}_i - A^*A\mathbf{x}_i + A^*Ae_i$. Therefore, we find our solution $\mathbf{r}_i^{LS} = A^*A\mathbf{e}_i$. *CQDF*

We can proceed by showing that $\mathbf{e}_i \perp_A \mathcal{K}_i(A^TA, \mathbf{r}_0^{LS})$ is also equivalent to $\mathbf{r}_i^{LS} \perp \mathcal{K}_i(A^TA, \mathbf{r}_0^{LS})$. If e_i is orthogonal to every vector in the Krylov subspace $\mathcal{K}_i(A^TA, \mathbf{r}_0^{LS})$ then by definition we have $\langle A\mathbf{v}, A\mathbf{e_i}\rangle = (Ae_i)^*(Av) = e_i^*A^*Ae_i = \langle \mathbf{v}, A^*A\mathbf{e_i}\rangle$ (by the properties of a norm and the conjugate transpose). Thus we can replace the equation of $\mathbf{r}_i^{LS}$ just found and we get $\langle \mathbf{v}, A^*A\mathbf{e_i}\rangle = \langle \mathbf{v}, \mathbf{r}_i^{LS}\rangle$ and it is equivalent to say that $\mathbf{r}_i^{LS} \perp \mathcal{K}_i(A^TA, \mathbf{r}_0^{LS})$. *CQFD*

Finally, we will show the following two simple lemmas:

Lemma 1. *Let $\mathbb{P}_{i,1}$ denote the space of polynomials of degree (at most) i with constant coefficient* 1*. For the CGLS iteration i it then holds that*

$$\mathbf{e}_i = q(A^TA)\mathbf{e}_0$$

with $q \in \mathbb{P}_{i,1}$.

Proof:

We know that

$$\mathcal{K}_i(A^TA, \mathbf{r}_0^{LS}) = \text{span}\{\mathbf{r}_0^{LS}, A^TA\mathbf{r}_0^{LS}, \ldots, (A^TA)^{i-1}\mathbf{r}_0^{LS}\}$$

and therefore that any vector in this subspace can be expressed as a polynomial p of degree i in A^TA applied to r_0^{LS}.

We know that $x_i - x_0$ lies in this Krylov subspace and thus $x_i - x_0 = p(A^TA)r_0^{LS}$ where p is polynomial of degree i. With the expressions found above we can rewrite it as: $e_i - e_0 = p(A^TA)A^TAe_0$.
Therefore, we find $e_i = p(A^TA)A^TAe_0 + e_0$ and because we know p is a polynomial of degree i-1 and we have a constant term e_0, we can express it with q, defined in the lemma, to find:

$$\mathbf{e}_i = q(A^TA)\mathbf{e}_0$$

CQFD

Lemma 2. *For any polynomial p and at any iterate i we have*

$$\mathbf{e}_i = p(A^TA)\mathbf{e}_0 \iff \mathbf{r}_i^{LS} = p(A^TA)\mathbf{r}_0^{LS}$$

Proof:
We can start by proving that if $\mathbf{e}_i = p(A^TA)\mathbf{e}_0$ then $\mathbf{r}_i^{LS} = p(A^TA)\mathbf{r}_0^{LS}$:
We can start by the equation found earlier $\mathbf{r}_i^{LS} = A^T(\mathbf{b} - A\mathbf{x}_i) = A^T\mathbf{b} - A^TA\mathbf{x}_i$ and $A^TAx_* = A^Tb$.
If we substitute it in $\mathbf{e}_i := \mathbf{x}_* - \mathbf{x}_i$, we can find that $\mathbf{r}_i^{LS} = A^TAe_i$. Knowing that $\mathbf{r}_0^{LS} = A^TAe_0$ and replacing e_i, we get $\mathbf{r}_i^{LS} = p(A^TA)\mathbf{r}_0^{LS}$.

Then we can prove that if $\mathbf{r}_i^{LS} = p(A^TA)\mathbf{r}_0^{LS}$ then $\mathbf{e}_i = p(A^TA)\mathbf{e}_0$:
We can replace $\mathbf{r}_0^{LS} = A^T\mathbf{b} - A^TA\mathbf{x}_0 = A^TAe_0$ (because $x_0 = 0$) in $\mathbf{r}_i^{LS} = p(A^TA)\mathbf{r}_0^{LS}$ to get $\mathbf{r}_i^{LS} = p(A^TA)A^TAe_0$. We also know that $\mathbf{r}_i^{LS} = A^TAe_i$ (found above) and thus $p(A^TA)A^TAe_0 = A^TAe_i$. We easily find our expression $\mathbf{e}_i = p(A^TA)\mathbf{e}_0$ back. *CQFD*

With these two lemmas we are now ready to prove the following important theorem:

Theorem 2. *At CGLS iteration k (at the end of the CGLS procedure) it holds that*

$$\mathbf{x}_* - \mathbf{x}_k = R_k(A^TA)(\mathbf{x}_* - \mathbf{x}_0)$$

with the Ritz polynomial R_k *given by*

$$R_k(t) := \prod_{j=1}^{k} \frac{\theta_j^{(k)} - t}{\theta_j^{(k)}},$$

in which $\theta_j^{(k)}$ denotes the jth Ritz value of A^TA i.e. the eigenvalues of $Q_k^*A^TAQ_k$, with Q_k the orthogonal basis of $\mathcal{K}_k(A^TA, \mathbf{r}_0^{LS})$.

Hint 1:Use that if θ is a zero of the polynomial $p(t)$, then $p(t) = (\theta - t)\tilde{p}(t)$, with $\tilde{p}$ one degree lower. In matrix polynomial form this reads as $p(t) = (\theta I_k - A^T A)\tilde{p}(A^T A)$.
Hint 2: Can you find a vector in your equations that can be written as $Q_k z$ for some z? (r_i^{LS} because stands in the Krylov space)
Hint 3: When are two polynomials that share the same zeros equal?: if hey have the same coefficients for corresponding terms

Proof:
We can see that the ritz polynomial is constructed in order to have zeros at the Ritz values. We know θ are the Ritz values of $A^T A$ and thus, using *Hint 1*, we will have this as follows:

$$p(A^T A) = (\theta I_k - A^T A)\tilde{p}(A^T A)$$

.

Using *Hint 2*, we will find a vector r_i^{LS} in the Krylov subspace that can be written as $Q_k z$ for some vector z and an orthogonal basis Q_k.

Using *Lemma 2*, we can find $r_k^{LS} = (\theta I_k - A^T A)\tilde{R}_k(A^T A)r_0^{LS}$, with the new polynomial $\tilde{R}_k$ without one ritz value. We will find that $\tilde{R}_k(A^T A)r_0^{LS}$ lies on the Krylov subspace and thus finding $\tilde{R}_k(A^T A)r_0^{LS} = Q_k z$ for a unique z.

Therefore, we find that $r_k^{LS} = (\theta I_k - A^T A)Q_k z$. If we multiply it by Q_k^*, we will find $Q_k^* r_k^{LS} = Q_k^*(\theta I_k - A^T A)Q_k z = 0$ (because r_k^{LS} is orthogonal to the Krylov subspace). Thus, we find that $Q_k^* A^T A Q_k a = \theta z$ for some vector z. For each θ_j^k true the iterations, we can write the characteristic polynomial, factoring out θ_j^k. We find

$$\tilde{R}_k(t) := \prod_{j=1}^{k-1} \frac{\theta_j^{(k)} - t}{\theta_j^{(k)}},$$

Now, we can use *Hint 3* and find that two polynomials have the same zeros if they have the same coefficients for corresponding terms. We see that R_k(degree k) and $\tilde{R}_k$(degree k-1) has the same (k-1) zeros(Ritz values). Thus,

with what we know we can easily find our Ritz polynomial as follow:

$$R_k(t) := \prod_{j=1}^{k} \frac{\theta_j^{(k)} - t}{\theta_j^{(k)}},$$

We know from the above expressions that we can write $r_k^L S = R_k(A^T A) r_0^{LS}$ and that $r_i^{LS} = A^T A e_i$. Thus, we find $e_k = R_k(A^T A) e_0$ and finally:

$$\mathbf{x}_* - \mathbf{x}_k = R_k(A^T A)(\mathbf{x}_* - \mathbf{x}_0)$$

with the Ritz polynomial R_k defined above. *CQFD*

If we have a matrix A, we know it singular values can be defined as σ_i. Conjugate gradient can be applied to normal equations $A^T Ax = A^T b$ and thus σ_i are the eigenvalues of $A^T A$. To have an easy access to the singular values we will make an SVD decomposition($A = U\Sigma V^T$):

$$V\Sigma^2 V^T x = V\Sigma U^T b$$

we simplified $U^T U = I$ and we used the fact that Σ is a diagonal square matrix.

From the Theorem we know

$$\mathbf{x}_* - \mathbf{x}_k = R_k(A^T A)(\mathbf{x}_* - \mathbf{x}_0) = R_k(V\Sigma^2 V^T) e_0$$

and thus we know that R_k has an effect on the error, a useful information in order to obtain the filter factors.
To get rid of V, we can decompose $e_0 = Vz$ such that z is a vector in the basis of V. We can write it as follow: $z_k = R_k(\Sigma^2) z$. We can see how R_k has an effect on each component z_i: $z_{ki} = R_k(\sigma_i^2) z_i$. The filter factor f_i is a measure of how much of the i-th component of the initial error passes through after k iterations and thus can be defined as:

$$f_i = 1 - R_k(\sigma_i^2)$$

We see that for larger singular values, $R_k(\sigma_i^2)$ is small, f_i is close to 1, meaning there is almost no filter (most of the error component is kept).
On the contrary for smaller singular values, $R_k(\sigma_i^2)$ is bigger, f_i is close to 0

(error component is damped). Small singular values correspond in the most case to the ill-posed part of the problem, so it seems quiet logic.
We see thus that the filter factors in the CG attenuate the error vector associated with the singular values and only keep the bigger singular values, keeping the most important informations about the solution. We see through those filter factors that it is regularized if the Ritz values converges to the eigenvalues of A^TA, in a decreasing order.

In the past methods the regularization parameter λ was explictly introduced to find a good solution. For the CG method we see that we used an implicit regularization parameter. Indeed, it will be regularized through the iteration number, named k, in order to have an early stopping and to prevent overfitting.

Note that the CG method does not include an L matrix! You should also be extremely careful with CG since the convergence can be delayed due to round-off errors in finite precision arithmetic.

2 Methods of choosing the regularization parameter

In this section we outline two methods for selecting the regularization parameter. Many of these are implemented in the package `regtools`, so you don't need to implement them. **Only for the CG method parameter selection methods are not implemented in `regtools`!**

2.1 The L-curve

The L-curve is a log-log plot with ($\log \|A\mathbf{x}_\lambda - \mathbf{b}\|_2$ represented on the veritcal axis and $\log \|L\mathbf{x}_\lambda\|_2$) on the horizontal axis (λ the continous regularization parameter). It is an interesting tool for choosing the regularization parameter based on a trade-off between the complexity of the regularized solution and the quality of the fit to the data. We will use the L-curve criterion for computing the curvature of the curve and seek the point with maximum curvature, defined as the L-corner. The L-corner is indeed an optimal point and thus a good balance for the trade-of: we will not have a too large or too low λ. It occurs when we have a maximum curvature, meaning the highest

change in trade-off. The L-curve criterion is already known to you from the theory. One is always interested in finding the 'corner' of the L-curve.
Here we will simply write code that generates the L-curve and find the optimal parameter by studying the resulting graph. I defined an ill-conditionned problem, to have more defined curvatures, and after computing an SVD I can calculate the regularized solution (we can choose any method other than CG). This will make us able to calculate the residual norm and the solution norm. This will represent the x- and y-axis of our L-curve plot. To find the maximum curvature of the l-curve, we don't have to forget that we are in a log-scale.
In general this can be done with regtools and this will be used in the last section.

2.2 Generalized cross-validation

GCV is based on the philosophy that if an arbitrary element $\mathbf{b}_i$ of the right-hand side $\mathbf{b}$ is left out, then the corresponding regularized solution should predict this observation well, and the choice of regularization parameter should be independent of an orthogonal transformation of $\mathbf{b}$. This (through some complex theory that you do not have to worry about) leads to choosing the regularization parameter, which minimizes the GCV function

$$G = \frac{\|A\mathbf{x}_{\mathrm{REG}} - \mathbf{b}\|_2^2}{(\mathrm{trace}(I_m - AA^I))^2}$$

where A^I is a matrix which produces the regularized solution $\mathbf{x}_{REG}$ when multiplied with $\mathbf{b}$, i.e. $\mathbf{x}_{\mathrm{REG}} = A^I\mathbf{b}$. For instance, in the case of Tikhonov regularization (with $L = I_m$) we have that $A^I = VF\Sigma^{-1}U^T$ (you can easily check this) when $A = U\Sigma V^T$ is the singular value decomposition of A and F is the corresponding diagonal matrix of filter factors at the parameter λ. Note that G is defined for both continuous and discrete regularization parameters. We have the following lemma:

Lemma 3. *We have that trace*$(I_m - AA^I) = m - (n - p) - \sum_{i=1}^{p} f_i$

Proof:
We remind that the trace is is the sum of the diagonal elements and that $\mathrm{trace}(A + B) = \mathrm{trace}(A) + \mathrm{trace}(B)$. Therefore we will split it into 2 calculations.

We know the trace(I_m) is equal to m (because it is an identity matrix). We also want to calculate the trace(AA^I) and therefore first find AA^I. Knowing from above that $A^I = VF\Sigma^{-1}U^T$ and thus that $AA^I = A(VF\Sigma^{-1}U^T) = U\Sigma V^T VF\Sigma^{-1}U^T$. Knowing that for an SVD, V is orthogonal ($V^T V = I_n$), we find $AA^I = U\Sigma F\Sigma^{-1}U^T$. We know that the diagonal elements of $\Sigma F\Sigma^{-1}$ are the filter factors f_i, for every singular value on Σ that is non zero. For the p non-zero singular values we have thus $\sum_{i=1}^{p} f_i$. For the n-p zero singular values on Σ, the inverse of those are normally not defined. Luckily we are in a regularization process, thus F will handle those values with his filter factors. In order that their effect remains neutral, the filter factors will put a contribution of 1 and thus the trace of this part is $(n-p)*1$. Therefore we find that trace(AA^I) = $(n-p) + \sum_{i=1}^{p} f_i$.
Thus: trace($I_m - AA^I$) = trace(I_m) − trace(AA^I) = $m - (n-p) - \sum_{i=1}^{p} f_i$
If we assume $x_0 = 0$ and $A^T A$ positive definite, n will represent the number of columns of A and thus because we know $A^T Ax = A^T b$, the dimension of our solution x. For p, we can consider it represent the rank of our matrix $A^T A$ and is thus related to number iterations before the convergence.

3 Application: the watchmaker and the bowl of soup

In this section we introduce a slightly silly application of the regularization methods described.

3.1 Context

Imagine you are a watchmaker, specializing in wristwatches. As such you have very sensitive hands and fingertips. In addition, you cannot allow them to be burned. Today, your partner has made you a bowl of soup, heated in the microwave and set it down at your workbench. You notice that the bowl is ceramic, which means it is probably very hot. Contrary, you deduce from the lack of steam that the soup is room temperature. Unfortunately, there is no cloth around that is thick enough to protect your hands from the scorching ceramic over the long trip back to the microwave. Fortunately though, you are skilled in integral equations and you can tell temperatures using your fingers with an accuracy of about $1e-3$ degrees. And so you put your finger in the soup, close to the bowl, but not touching, and measure the

temperature as a function of time. You use this to determine the temperature of the bowl by solving *the inverse heat equation.*

3.2 The inverse heat equation

We model this as the inverse of the following heat equation:

$$\begin{aligned} \frac{\partial}{\partial t}T &= \frac{\partial^2}{\partial^2 x}T \\ T(x,0) &= 0 \\ T(0,t) &= f(t) \end{aligned}$$

for $0 < x < \infty$ and $0 \leq t < \infty$. Here, the temperature $T = 0$ actually corresponds to room temperature, that is 20 degrees Celsius. The origin ($x = 0$) corresponds to the bowl. The variable t in our example is measured in minutes. By inverse we mean: while the above calculates T given the boundary condition f , we wish to compute f from the observed $T(y,t) := g(t)$ at a distance y from the origin. It is known that this is given by

$$Kf = g$$

with

$$(Kf)(t) = \frac{y}{2\sqrt{\pi}} \int_0^t \frac{f(\xi)}{(t-\xi)^{3/2}} \exp(\frac{-y^2}{4(t-\xi)})\, d\xi.$$

This is a Volterra integral equation, but can be transformed into a Fredholm equation by extension. It is thus immediately suspect: the discretization of this problem is probably ill-posed.

3.3 Provided data

You have been provided a matrix K, obtained by the mid-point rule discretization of the operator K. In addition you are given an exact solution f and a perturbed (random normally distributed noise, $\sigma = 1e-3$) right-hand-side $\mathbf{g}$. Both are given over a time range of 10 minutes.

3.4 Extra information about the solution

Since regularization is all about using information about the solution, I will now provide some information that you can use. You can verify this by

looking at the exact solution given. **Not every piece of information needs to be used, or might even be helpful!**

(1) The solution is not oscillatory. This can be achieved by requiring that its second derivative is small

(2) The solution is eventually zero. In fact, if $t > 5$, f is negligible

(3) The temperature is always positive i.e. $f > 0$

(4) Initially, the temperature is stationary, that is, $g'(0) = 0$

3.5 Implementation

First of all, we want to know if our initial problem is ill-posed. We can indeed see that the condition number of my matrix is $1.115683986260964e{+}124$ which is huge, conducting to an ill-posed problem. Indeed, the definition of a condition number is given by: "function measures how much the output value of the function can change for a small change in the input argument". When we try to resolve it 'naively' using *mldivide* we see that it give us an solution with only *Nan* values, still comfirming our problem is ill-posed.
Luckily, we have seen in this report how to handle ill-posed problems. We are given the true solution f, which we will use to compare each of the solutions. Firstly, we could use a 'naive' method by minimizing the regularization problem described in section 1 and then use the *mldivide* to solve it. However, this method is not optimal if we don't know our optimal regularization parameter λ. For finding the optimal λ we will compare the two method,l-curve and the Generalized Cross-Validation(GCV), for each regularization method.

- We can start by the **Tikhonov** regularization. We have two kind

Tikhonov Classical: for $L = I_m$

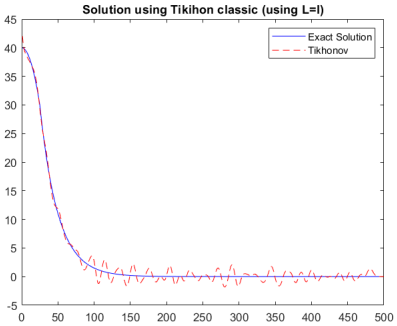

Figure 2: Using l-curve

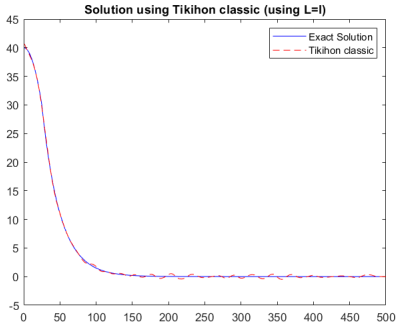

Figure 3: Using GCV

We wee that the regularization method computed, in the two cases, gives us a solution that is relatively close to the true solutions. However, we see more oscillations when using the l-curve and thus the solution using the optimal lambda with GCV is smoother. This is logic because we know the GCV

method will try to have the smoother solution and lcurve will try to fit the data better. Here we will prefer using the regularization parameter obtained with GCV.

Tikhonov Advanced: for L not equal to I_m. I chose a second derivative L such that it penalizes oscillations.

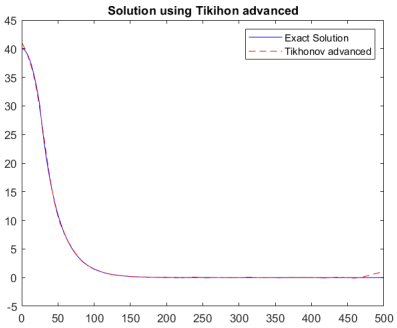

Figure 4: Using l-curve

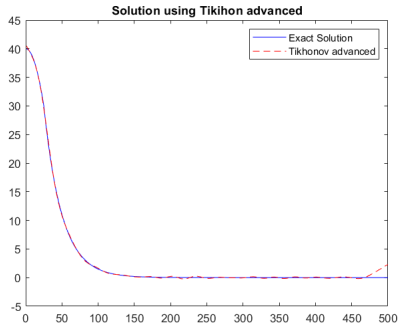

Figure 5: Using GCV

Here we see that the solution still almost entirely fit the true solution but with less oscillations due to the choice of L that will penalize the oscillations. In addition, we can see that choosing either lcurve or either GCV is a good choice.

- Then we will have the **SVD-based methods**:

We can start with the **TSVD** regularization

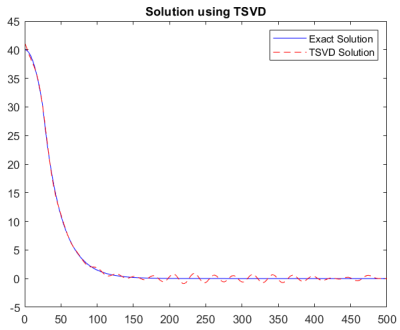

Figure 6: Using l-curve

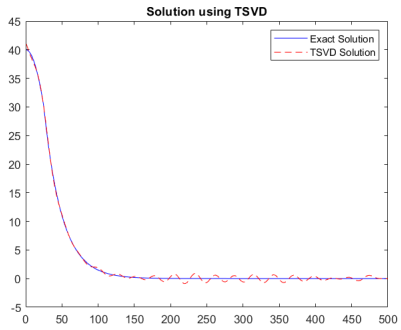

Figure 7: Using GCV

Again, it is close to the true solution, but this time with a little bit more oscillations around 0. The choice of singular values that we kept to calculate the solution is maybe a little bit too much, leading to this oscillations. But sometimes it is impossible to avoid oscillations so we can conclude that is relatively a good fit.

For the **DSVD** regularization we have:

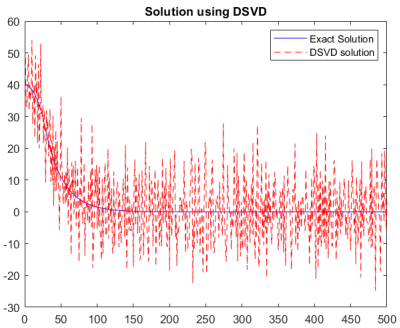

Figure 8: Using l-curve

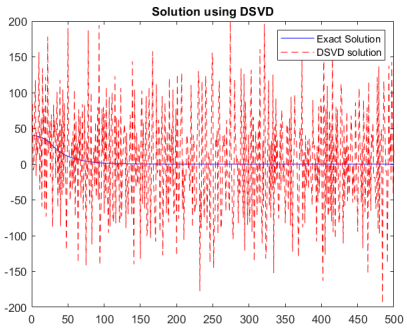

Figure 9: Using GCV

Here we can see that we have significant oscillations, which will make us severly deviate from our true solution. The parameter selection(number of singular values to keep) is the problem in this case. Indeed, in this method L is considered as I and thus we could not apply a condition on the second

derivative to avoid oscillations. We will see that we used an additional condition with DGSVD to avoid those oscillations.

For the **TGSVD** regularization we have:

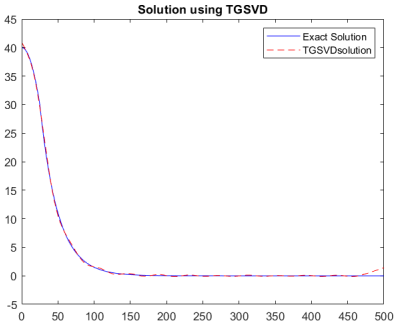

Figure 10: Using l-curve

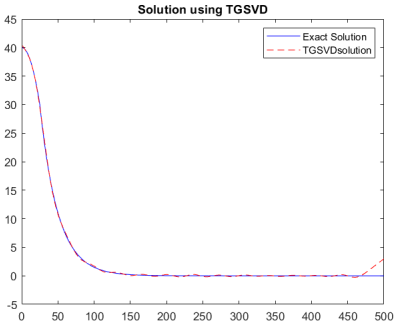

Figure 11: Using GCV

The TGSVD, with L a second derivative, is a good regularization method that fits quiet well the good solution with less oscillations than TSVD. However, we see that at the end, we have a little deviation. The information at the end is lost due to the truncation number. However the deviation is less there when using l-curve and the amount information lost is small compared to the entire solution. Therefore using the optimal parameter with lcurve is a good choice.

And then for the **DGSVD** regularization we have:

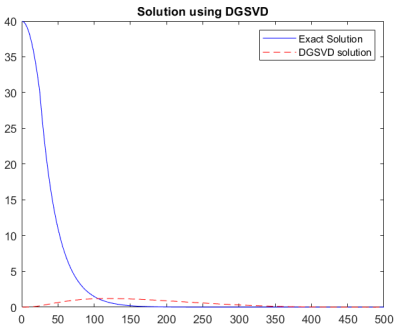

Figure 12: Using l-curve

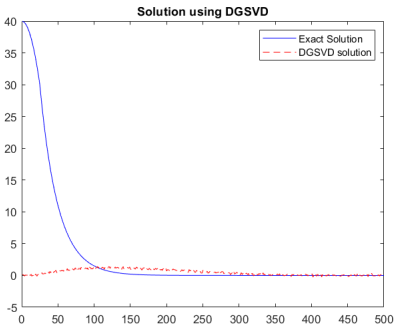

Figure 13: Using GCV

Compared to DSVD they are less oscillations but at the cost of another problem. Indeed, we can see that we loose a lot of informations about the beginning of the true solution. I tried to use semi-norm matrices to impose a boundary condition at t=0, but I failed to implement it.

- Finally we will have the **Conjugate Gradient based** regularization.

Like said in the other sections, the regularization parameter is now implicitly defined in the CG algorithm by the number of iterations k. Therefore I plot the CG solution with different values of k:

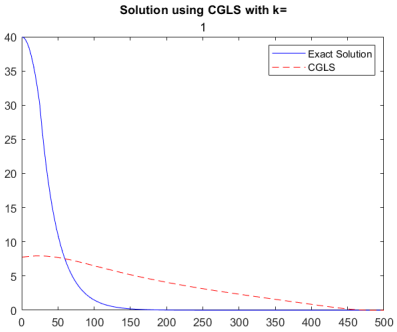

Figure 14: CG with k=1

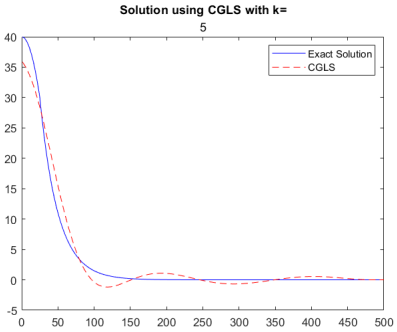

Figure 15: CG with k=5

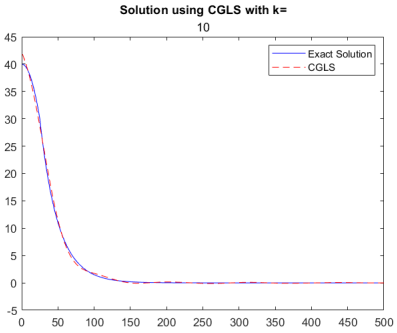

Figure 16: CG with k=10

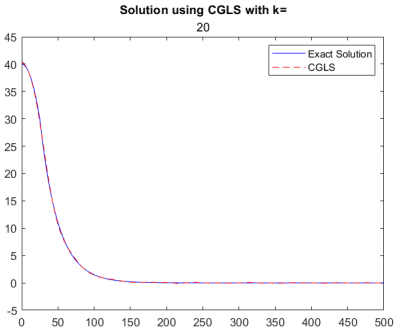

Figure 17: CG with k=20

We see that when k increase, the CG algorithm tends to converge to the exact solution. This is logic because as the size of the subspace increase, the approximate solution become closer to the true solution. When k=1, the CG algorithm will filter the information in the direction of the largest singular value, missing some important information to fit well the true solution. Taking a k too large is also not a good solution, because we will take very small singular values that include information that may not be very important and thus adding some noise (oscillations). We can conclude by saying that choosing k=20 is a good choice (taking a larger k does not really improve our approximation).

4 Conclusion

In conclusion, we see that having an ill-posed problem is not a problem due to the regularization methods that make us able to find back a globally good approximation of our true solution. However, choosing the right method is crucial for finding a right fitting solution. In our case, with the ill-posed heat equation we will prefer using advanced methods with the involving of an L matrix other than identity to be able to penalize some deviations from the true solution. The conjugate gradient is also a good regularization method. For the choice of optimal regularization parameter, lcurve and GCV technique both give globally good results for the heat equation. They have both different goals: prioritize fitting the data (lcurve) or maintaining the smoothness(gcv). Depending on the problem we will perfer either one or the other.

5 References

General: https://arxiv.org/pdf/1801.09922.pdf
Chatgpt to reformulate some sentences

Tikihinov: https://fr.wikipedia.org/wiki/R%C3%A9gularisation_de_Tikhonov
https://www.sciencedirect.com/science/article/abs/pii/S016971611830021X
L-curve: https://www.sintef.no/globalassets/project/evitameeting/2005/lcurve.pdf/

Generalized cross-validation: https://arxiv.org/pdf/2311.11442.pdf

Regtools manuel: http://www2.compute.dtu.dk/~pcha/Regutools/RTv4manual.pdf