Answers for homework assignment 3

Problem 1 (Gaussian elimination). The computation is long and tedious and isn’t repeated here. The result is

\[ x = \begin{pmatrix} -64 \\ 900 \\ -2520 \\ 1820 \end{pmatrix} \]

The fact that the computation was tedious was the point of this exercise – it’s not that hard to find the solution of a $3 \times 3$ system on a piece of paper, but it’s much more work to do the same for a $4 \times 4$ system. This illustrates the difficulty with Gaussian elimination: it’s run-time just grows unacceptably quickly.

The Hilbert matrix is pretty badly conditioned. The $4 \times 4$ matrix already has a condition number $\kappa_\infty = 28375$, and this is growing very quickly as we increase the size: for the $6 \times 6$ matrix, we have $\kappa_\infty = 2.9 \cdot 10^7$ and for $10 \times 10$ the condition number is $\kappa_\infty = 3.5 \cdot 10^{13}$. With such a large condition number, numerical solution of linear systems with these matrices become very unstable.

Problem 2 (Gaussian elimination). There are a number of ways to compute the inverse. One way is to solve for the vectors $z_i$ that satisfy the set of $n$ equations

\[ Az_i = e_i, \]

where $e_i$ are the unit vectors with a single one in row $i$ and zeros everyone else. If one has all these vectors $z_i$, it is easy to see that $z_i$ is the $i$th column of $A^{-1}$. (This is so, because $z_i = A^{-1}e_i$, and the multiplication of a matrix, here $A^{-1}$ with the $i$th unit vector yields its $i$th column – do the experiment with a matrix $X$ for which you have all the entries and multiply it with $e_1, e_2, \ldots$)

A more elegant way is to compute the LU decomposition $A = LU$, then $A^{-1} = (LU)^{-1} = U^{-1}L^{-1}$. $L$ and $U$ are both triangular matrices for which it is simple to determine the inverses.

For the present matrix, the inverse is

\[ A^{-1} = \begin{pmatrix} 16 & -120 & 240 & -140 \\ -120 & 1200 & -2700 & 1680 \\ 240 & -2700 & 6480 & -4200 \\ -140 & 1680 & -4200 & 2800 \end{pmatrix}. \]
Problem 3 (Gaussian elimination). The process shown in class breaks down because there is a zero entry on the diagonal, by which we would have to divide. That doesn’t work, of course.

However, remember that there are three things you can do to a linear system that leave the solution unchanged: (i) multiplication of a single equation by a scalar, (ii) addition of a multiple of one equation to another equation, and (iii) exchanging two equations. The Gaussian elimination process, just as the LU decomposition, only used rules (i) and (ii). In order to make the system in the problem solvable, we have to use the third rule, by sorting the equations in order (2), (3), (1). This yields the same matrix as at the beginning of the Problem, though with a different right hand side. The matrix is now in a form that is readily invertible.

The process of picking an order of the equations is usually called “pivoting”.

Problem 4 (Multidimensional root finding). Let us first take a look at the two functions:

The first function obviously has a lot of local minima, though we don’t know right away which one is the global minimum. The Newton iteration will converge to one of the local minima, most likely not the global one. (In fact, the following is easy to see: the minima of the sum of the two cosine functions are on a regular grid, and have values of -2. But then $e^x$ is also added, adding a positive value to the -2. The further we go to the left, the smaller $e^x$ is going to be. The global minimum would therefore be somewhere at $x = -\infty$.)

The second function does not have a single, isolated minimum, but rather a whole line that runs at the bottom of the curvy valley along which the function attains its minimal valley. This is going to cause trouble later on.

The Newton iteration for this problem reads

$$X_{k+1} = X_k + \delta X,$$
$$[\nabla F(X_k)] \delta X_k = -F(X_k),$$

where $F = (f_1, f_2)$. Putting this all into a program leads us to converge against $X = (-3.156, -3.156)$ for part a).
For part b), it is mostly up to luck where exactly the iteration ends, but it should be on the minimal line \( y = \sin(4x) \), since each point of this line is a minimum. (This is easy to see: we can write \( g(x, y) = e^a + e^{-a} = \cosh a \), where \( a = y - \sin(4x) \); the minimym of \( g(x, y) \) is therefore attained whenever \( a = 0 \), i.e. for \( y = \sin(4x) \).) Once we reach this line, we must have \( F(X) = 0 \), since this is a minimum, but in contrast to part a), we have that \( \nabla F(X) \) is a singular matrix at all points \( X \) of this line, i.e. we can’t invert \( \nabla F(X) \) to obtain the next iteration. If this happens in practice and no precautions are taken, then following iterations simply show \( NaN \) or similar as answer. This indicates that the problem is degenerate: there is not a single isolated minimum, but a continuous line of points along which the function has minimal value.

Finally, the starting point \( X_0 = (0, 0) \) already lies on this line of minima. Therefore even the first step doesn’t work, since \( \nabla F(X_0) \) is not invertible. In fact,

\[
\nabla F(X_0) = \begin{pmatrix} 16 & -4 \\ -4 & 1 \end{pmatrix},
\]

which is a matrix that has a zero determinant (one of the rows is a multiple of the other).