Problem Set 6: Detrending Kepler Data, part III

1. Based on the fact that it was taking over an hour to not even begin to converge for the Gauss-Seidel method that I used throughout this assignment (so it does work), I only have values for the SOR method with a pre-conditioned matrix.

Method	**N = 2 Solution**	**Time (s)**	**Steps**	**Tol.**
Normal eq.	y =(2.52985e+05)+x(-2.29530e+02)+x^2(2.88217e-01)	5.064e-4	3	N/A
QR	y =(2.52985e+05)+x(-2.29530e+02)+x^2(2.88217e-01)	1.543e-4	3	N/A
SOR	y =(2.52998e+05)+x(-2.29619e+02)+x^2(2.88365e-01)	49.642	105760	1e-8

The behavior of the direct solvers for the first 10 polynomials is described in Figure 1.

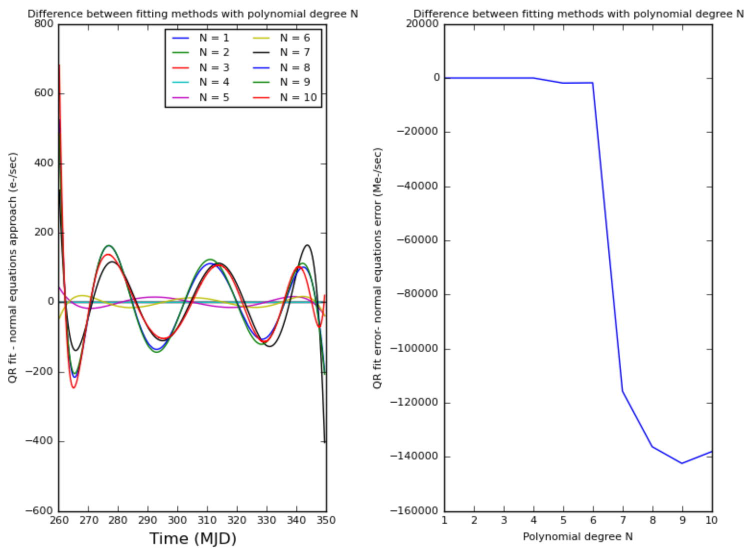

Fig. 1.— Behavior of direct solvers on unconditioned data.

2. Preconditioning the data matrix led to a rapid decrease in the number of steps and required to achieve a high-fidelity solution. The difference was about 3 orders of magnitude for the SOR method. The condition number $\kappa(A)$ went from 1.460e7 to 23.02 with preconditioning and $\kappa(A^TA)$ went from 2.133e15 to 513.

Method	**N = 2 Solution**	**Time (s)**	**Steps**	**Tol.**
Normal eq.	y =(2.12770e+05)+x(-7.09679e+03)+x^2(2.29583e+03)	5.064e-4	3	N/A
QR	y =(2.12770e+05)+x(-7.09679e+03)+x^2(2.29583e+03)	1.543e-4	3	N/A
Gauss-Seidel	y =(2.12771e+05)+x(-7.09736e+03)+x^2(2.29639e+03)	5.03e-2	1032	1e-8
SOR	y =(2.12771e+05)+x(-7.09475e+03)+x^2(2.29391e+03)	3.55e-2	527	1e-8

The behavior of the direct solvers for the first 10 polynomials on the preconditioned data is described in Figure 2.

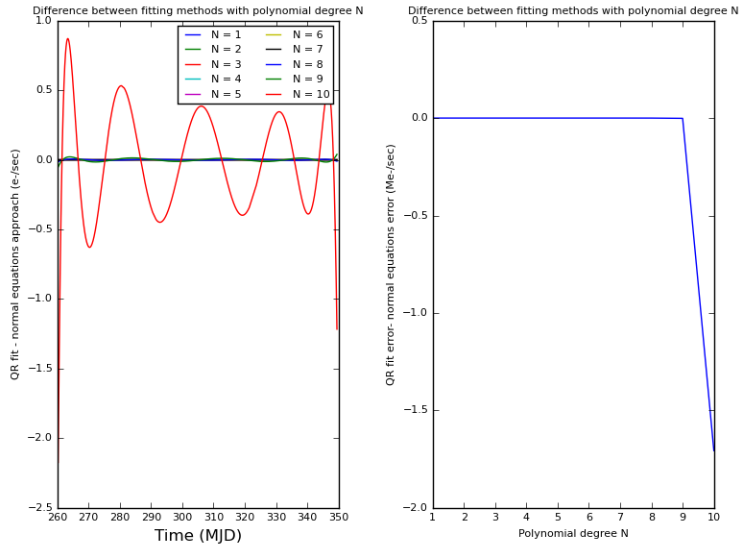

Fig. 2.— Behavior of direct solvers on preconditioned data.

3. The convergence of both of these methods is described in Figure 3. Adding another degree to the polynomial caused a 20x slow down for the Gauss-Seidel method and 5x slow down for SOR. The effect on the direct solvers was approximately null.

Method	**N = 3 Solution**	**Time (s)**	**Steps**	**Tol.**
Normal eq.	y =(2.12425e+05)+x(-2.87546e+03)+x^2(-8.31596e+03)+x^3(7.08335e+03)	5.748e-4	4	N/A
QR	y =(2.12425e+05)+x(-2.87546e+03)+x^2(-8.31596e+03)+x^3(7.08335e+03)	1.802e-4	4	N/A
Gauss-Seidel	y =(2.12426e+05)+x(-2.87626e+03)+x^2(-8.31399e+03)+x^3(7.08206e+03)	1.468	18763	1e-9
SOR	y =(2.12426e+05)+x(-2.87669e+03)+x^2(-8.31350e+03)+x^3(7.08197e+03)	0.1842	3494	1e-8

4. Figure 3 shows the convergence of all 4 iterative methods. Conjugate is the fastest by far, and steepest descent is almost 20x slower than the Gauss-Seidel method.

Method	**N = 3 Solution**	**Time (s)**	**Steps**	**Tol.**
Normal eq.	y =(2.12425e+05)+x(-2.87546e+03)+x^2(-8.31596e+03)+x^3(7.08335e+03)	5.748e-4	4	N/A
QR	y =(2.12425e+05)+x(-2.87546e+03)+x^2(-8.31596e+03)+x^3(7.08335e+03)	1.802e-4	4	N/A
Gauss-Seidel	y =(2.12426e+05)+x(-2.87626e+03)+x^2(-8.31399e+03)+x^3(7.08206e+03)	1.468	18763	1e-9
SOR	y =(2.12426e+05)+x(-2.87669e+03)+x^2(-8.31350e+03)+x^3(7.08197e+03)	0.1842	3494	1e-8
Conjugate gradient	y =(2.12423e+05)+x(-2.87389e+03)+x^2(-8.31392e+03)+x^3(7.08535e+03)	6.71e-4	15	1e-4
Steepest descent	y =(2.12426e+05)+x(-2.87625e+03)+x^2(-8.31304e+03)+x^3(7.08210e+03)	20.958	55256	1e-9

5. Iterative methods vary from being very fast to very slow. While it would always be preferable, based solely on the data we have here, to use a direct solve, the conjugate gradient comes very close to being on the same timescale as direct solves, and therefore would be ideal for problems that must be solved iteratively.

Code used in this assignment can be found at https://github.com/brbordwell/ASTR_5540/HW6

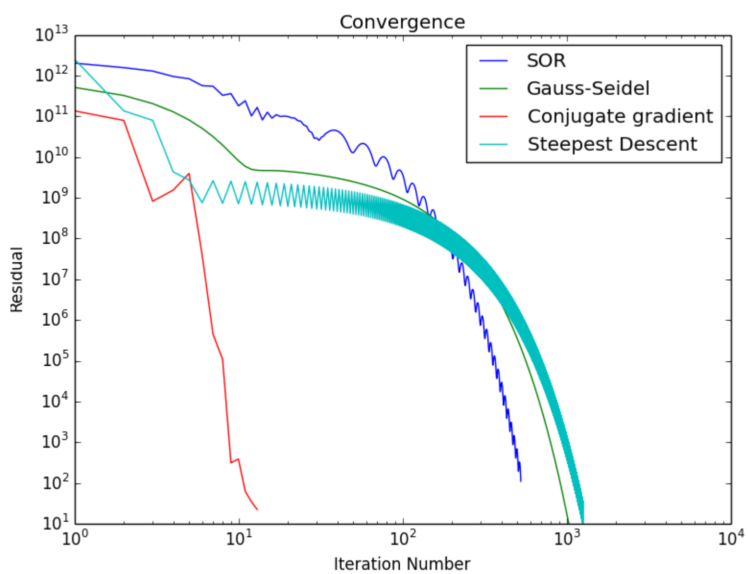

Fig. 3.— Convergence of our iterative methods for N=3