# CS5321 Numerical Optimization Homework 3

Due Nov 27

1. (30%) For the linear least square problem, most people use polynomials as the basis, $1, x, x^2, \cdots$. If we use the trigonometric functions as basis, we have Fourier approximation. For example, the model function is

$$f(x) = \frac{a_0}{2} + a_1 \cos(x) + \cdots + a_m \cos(mx) + b_1 \sin(x) + \cdots + b_m \sin(mx),$$

and the measured data are $(x_k, y_k)$, where $x_k = \frac{k2\pi}{n}$ for $k = 0, \cdots, n-1$.

   (a) Write down it as an optimization problem $\min_{\vec{x}} \|A\vec{x} - \vec{b}\|$, where $\vec{x} = (a_0, a_1, \ldots, a_m, b_1, \ldots, b_m)^T$, and $\vec{b} = (y_0, y_1, \ldots, y_n - 1)^T$.

   (b) Show that each column vector of $A$ is orthogonal to each other.

   (c) Use the normal equation to find the solution of $a_i$ and $b_i$.

   (d) Comparing to the polynomial basis, what is the advantage of using trigonometric basis?

2. (70%) Implement the simplex method for linear programming. The pseudo code is in Figure 1. The calling interface will be like

```
[x, case] = mysimplex(c, A, b, x0)
```

which solves

$$\begin{aligned} \min_{\vec{x}} \quad & \vec{c}^T \vec{x} \\ \text{subject to} \quad & A\vec{x} = \vec{b} \\ & \vec{x} \geq 0 \end{aligned}$$

The return value `case` should be 0, 1, or 2, which means (0) solved, (1) unbounded, (2) infeasible. You can assume $\vec{x}_0$ is a feasible point. Also, print out each $\vec{x}_i$ during the computation. Use it to solve the following problem.

$$\begin{array}{ll} \max_{x_1,x_2} & z = 8x_1 + 5x_2 \\ \text{s.t.} & 2x_1 + x_2 \le 1000 \\ & 3x_1 + 4x_2 \le 2400 \\ & x_1 + x_2 \le 700 \\ & x_1 - x_2 \le 350 \\ & x_1, x_2 \ge 0 \end{array}$$

---

(1) Given a basic feasible point $\vec{x}_0$ and the corresponding index set $\mathcal{B}_0$ and $\mathcal{N}_0$.

(2) For $k = 0, 1, \ldots$

(3) Let $B_k = A(:, \mathcal{B}_k), N_k = A(:, \mathcal{N}_k)$, $\vec{x}_B = \vec{x}_k(\mathcal{B}_k), \vec{x}_N = \vec{x}_k(\mathcal{N}_k)$, and $\vec{c}_B = \vec{c}_k(\mathcal{B}_k), \vec{c}_N = \vec{c}_k(\mathcal{N}_k)$.

(4) Compute $\vec{s}_k = \vec{c}_N - N_k^T B_k^{-1} \vec{c}_B$ (pricing)

(5) If $\vec{s}_k \ge 0$, return the solution $\vec{x}_k$. (found optimal solution)

(6) Select $q_k \in \mathcal{N}_k$ such that $\vec{s}_k(i_q) < 0$, where $i_q$ is the index of $q_k$ in $\mathcal{N}_k$

(7) Compute $\vec{d}_k = B_k^{-1} A_k(:, q_k)$. (search direction)

(8) If $\vec{d}_k \le 0$, return `unbounded`. (unbounded case)

(9) Compute $[\gamma_k, i_p] = \min_{i, \vec{d}_k(i) > 0} \dfrac{\vec{x}_B(i)}{\vec{d}_k(i)}$ (ratio test)

(The first return value is the minimum ratio; the second return value is the index of the minimum ratio.)

(10) $$x_{k+1} \begin{pmatrix} \mathcal{B} \\ \mathcal{N} \end{pmatrix} = \begin{pmatrix} \vec{x}_B \\ \vec{x}_N \end{pmatrix} + \gamma_k \begin{pmatrix} -\vec{d}_k \\ \vec{e}_{i_q} \end{pmatrix}$$

($\vec{e}_{i_q} = (0, \ldots, 1, \ldots, 0)^T$ is a unit vector with $i_q$th element 1.)

(11) Let the $i_p$th element in $\mathcal{B}$ be $p_k$. (pivoting)

$\mathcal{B}_{k+1} = (\mathcal{B}_k - \{p_k\}) \cup \{q_k\}$, $\mathcal{N}_{k+1} = (\mathcal{N}_k - \{q_k\}) \cup \{p_k\}$

---

Figure 1: The simplex method for solving (minimization) linear programming