IPM forumalations

Pratyai Mazumder

March 14, 2023

Contents

1 Original Problem

1.i Primal

$$\begin{aligned} \min_{\mathbf{x}} \quad & \mathbf{c}^T\mathbf{x} \\ \text{s.t.} \quad & \mathbf{A}\mathbf{x} = \mathbf{b}, \\ & \mathbf{x} \leq \boldsymbol{u}, \\ & \mathbf{x} \in \mathbb{R}^m_{\geq 0} \end{aligned} \tag{1}$$

1.ii Primal Standard Form

$$\begin{aligned} \min_{\mathbf{x}} \quad & \mathbf{c}^T\mathbf{x} \\ \text{s.t.} \quad & \mathbf{A}\mathbf{x} = \mathbf{b}, \\ & \mathbf{x} + \mathbf{x}_u = \boldsymbol{u}, \\ & \mathbf{x}, \mathbf{x}_u \in \mathbb{R}^m_{\geq 0}, \mathbb{R}^m_{\geq 0} \end{aligned} \tag{2}$$

1.iii Dual

$$\begin{aligned} \max_{\mathbf{y}, \mathbf{y}_u} \quad & \mathbf{b}^T\mathbf{y} + \mathbf{u}^T\mathbf{y}_u \\ \text{s.t.} \quad & \mathbf{A}^T\mathbf{y} + \mathbf{y}_u \leq \mathbf{c}, \\ & \mathbf{y}, \mathbf{y}_u \in \mathbb{R}^n, \mathbb{R}^m_{\geq 0} \end{aligned} \tag{3}$$

1.iv Dual Standard form

$$\begin{aligned} \max_{\mathbf{y},\mathbf{y}_u} \quad & \mathbf{b}^T\mathbf{y} + \mathbf{u}^T\mathbf{y}_u \\ \text{s.t.} \quad & \mathbf{A}^T\mathbf{y} + \mathbf{y}_u + \mathbf{z} = \mathbf{c}, \\ & \mathbf{y}, \mathbf{y}_u, \mathbf{z} \in \mathbb{R}^n, \mathbb{R}^m_{\geq 0}, \mathbb{R}^m_{\geq 0} \end{aligned} \tag{4}$$

But if we want $\mathbf{y}$ variables to be free, then dualize the standard from, then standardize again:

$$\begin{aligned} \max_{\mathbf{y},\mathbf{y}_u} \quad & \mathbf{b}^T\mathbf{y} + \mathbf{u}^T\mathbf{y}_u \\ \text{s.t.} \quad & \mathbf{A}^T\mathbf{y} + \mathbf{y}_u + \mathbf{z}_1 = \mathbf{c}, \\ & \mathbf{y}_u + \mathbf{z}_2 = \mathbf{0}, \\ & \mathbf{y}, \mathbf{y}_u, \mathbf{z}_1, \mathbf{z}_2 \in \mathbb{R}^n, \mathbb{R}^m, \mathbb{R}^m_{\geq 0}, \mathbb{R}^m_{\geq 0} \end{aligned} \tag{5}$$

2 Solver Forms

2.i Long Step Path Following Method

Ref: Numerical Optimization (Alg. 14.2)

It turns out that staying inside the *good neighbourhood* of the central path can be difficult with an approximate solver. So, we will use `Mehrotra's Predictor Corrector Method` to solve the sample problems. The formulation is the same in both cases, so we will describe it here for simplicity.

Due to the formulation, we have to work with the standard form.

2.i.1 Primal

$$\begin{aligned} \min \quad & c^T x \\ \text{s.t.} \quad & Ax = b, \\ & x \in \mathbb{R}^n_{\geq 0} \end{aligned} \tag{6}$$

$$A = \begin{bmatrix} \mathbf{A} & \mathbf{0} \\ \mathbf{I} & \mathbf{I} \end{bmatrix} \quad | \quad x = \begin{bmatrix} \mathbf{x} \\ \mathbf{x}_u \end{bmatrix} \quad | \quad b = \begin{bmatrix} \mathbf{b} \\ \mathbf{u} \end{bmatrix} \quad | \quad c = \begin{bmatrix} \mathbf{c} \\ \mathbf{0} \end{bmatrix}$$

2.i.2 Dual

$$\begin{aligned} \max \quad & b^T y \\ \text{s.t.} \quad & A^T\lambda + s = c, \\ & \lambda, s \in \mathbb{R}^m, \mathbb{R}^n_{\geq 0} \end{aligned} \tag{7}$$

$$A = \begin{bmatrix} \mathbf{A} & \mathbf{0} \\ \mathbf{I} & \mathbf{I} \end{bmatrix} \quad | \quad \lambda = \begin{bmatrix} \mathbf{y} \\ \mathbf{y}_u \end{bmatrix} \quad | \quad b = \begin{bmatrix} \mathbf{b} \\ \mathbf{u} \end{bmatrix} \quad | \quad c = \begin{bmatrix} \mathbf{c} \\ \mathbf{0} \end{bmatrix} \quad | \quad s = \begin{bmatrix} \mathbf{z}_1 \\ \mathbf{z}_2 \end{bmatrix} = \begin{bmatrix} \mathbf{s} \\ \mathbf{s}_u \end{bmatrix}$$

2.i.3 Big KKT

$$\begin{bmatrix} 0 & A^T & I \\ A & 0 & 0 \\ S & 0 & X \end{bmatrix} \begin{bmatrix} \delta_x \\ \delta_y \\ \delta_s \end{bmatrix} = \begin{bmatrix} -r_d \\ -r_p \\ -r_c \end{bmatrix}$$

2.i.4 Small KKT

$$\begin{bmatrix} -X^{-1}S & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} \delta_x \\ \delta_y \end{bmatrix} = \begin{bmatrix} -r_d + X^{-1}r_c \\ -r_p \end{bmatrix}$$
$$\delta_s = -X^{-1}(r_c + Sd_x)$$

2.i.5 No KKT

Let $M := AS^{-1}XA^T$ (note: this is *not necessarily* a Laplacian). Then,

$$\begin{aligned} \delta_y &= M^+(-r_p - AS^{-1}Xr_d + AS^{-1}r_c) \\ \delta_x &= -S^{-1}(r_c - X(r_d + A^Td - y)) \\ \delta_s &= -X^{-1}(r_c + Sd_x) \end{aligned}$$

Observe that solving the $Md_y = -r_p - AS^{-1}Xr_d + AS^{-1}r_c$ system is difficult — even a fast Laplacian solver cannot directly help. The other inversions of diagonal matrices and multiplications with sparse or diagonal matrices can be done relatively easily (i.e. $\mathcal{O}(nnz)$).

2.i.6 Approximate KKT

Substituting in the $Md_y = -r_p - AS^{-1}Xr_d + AS^{-1}r_c = -r_p - AS^{-1}(Xr_d - r_c)$ system:

$$\begin{bmatrix}\mathbf{A} & 0\\ I & I\end{bmatrix}\begin{bmatrix}\mathbf{S}^{-1} & 0\\ 0 & \mathbf{S}_u^{-1}\end{bmatrix}\begin{bmatrix}\mathbf{X} & 0\\ 0 & \mathbf{X}_u\end{bmatrix}\begin{bmatrix}\mathbf{A}^T & I\\ 0 & I\end{bmatrix}\begin{bmatrix}d_y\\ d_{yu}\end{bmatrix} = -\begin{bmatrix}r_p\\ r_{pu}\end{bmatrix} - \begin{bmatrix}\mathbf{A} & 0\\ I & I\end{bmatrix}\begin{bmatrix}\mathbf{S}^{-1} & 0\\ 0 & \mathbf{S}_u^{-1}\end{bmatrix}\left(\begin{bmatrix}\mathbf{X} & 0\\ 0 & \mathbf{X}_u\end{bmatrix}\begin{bmatrix}r_d\\ r_{du}\end{bmatrix} + \begin{bmatrix}r_c\\ r_{cu}\end{bmatrix}\right)$$

Simplifying:

$$\begin{bmatrix}\mathbf{A}\mathbf{S}^{-1}\mathbf{X}\mathbf{A}^T & \mathbf{A}\mathbf{S}^{-1}\mathbf{X}\\ \mathbf{S}^{-1}\mathbf{X}\mathbf{A}^T & \mathbf{S}^{-1}\mathbf{X} + \mathbf{S}_u^{-1}\mathbf{X}_u\end{bmatrix}\begin{bmatrix}d_y\\ d_{yu}\end{bmatrix} = \begin{bmatrix}-r_p - \mathbf{A}\mathbf{S}^{-1}(\mathbf{X}r_d - r_c)\\ -r_{pu} - \mathbf{S}^{-1}(\mathbf{X}r_d - r_c) - \mathbf{S}_u^{-1}(\mathbf{X}_u r_{du} - r_{cu})\end{bmatrix}$$

Let $\mathbf{K} := \mathbf{S}^{-1}\mathbf{X} + \mathbf{S}_u^{-1}\mathbf{X}_u$ and $r_q := -r_{pu} - \mathbf{S}^{-1}(\mathbf{X}r_d - r_c) - \mathbf{S}_u^{-1}(\mathbf{X}_u r_{du} - r_{cu})$:

$$\begin{bmatrix}\mathbf{A}\mathbf{S}^{-1}\mathbf{X}\mathbf{A}^T & \mathbf{A}\mathbf{S}^{-1}\mathbf{X}\\ \mathbf{S}^{-1}\mathbf{X}\mathbf{A}^T & \mathbf{K}\end{bmatrix}\begin{bmatrix}d_y\\ d_{yu}\end{bmatrix} = \begin{bmatrix}-r_p - \mathbf{A}\mathbf{S}^{-1}(\mathbf{X}r_d - r_c)\\ r_q\end{bmatrix}$$

Observing that $\mathbf{K}$ is invertible, and scaling:

$$\begin{bmatrix}\mathbf{A}\mathbf{S}^{-1}\mathbf{X}\mathbf{A}^T & \mathbf{A}\mathbf{S}^{-1}\mathbf{X}\\ \mathbf{K}^{-1}\mathbf{S}^{-1}\mathbf{X}\mathbf{A}^T & I\end{bmatrix}\begin{bmatrix}d_y\\ d_{yu}\end{bmatrix} = \begin{bmatrix}-r_p - \mathbf{A}\mathbf{S}^{-1}(\mathbf{X}r_d - r_c)\\ \mathbf{K}^{-1}r_q\end{bmatrix}$$

Eliminating d_{yu}:

$$\begin{bmatrix}\mathbf{A}(\mathbf{S}^{-1}\mathbf{X} - \mathbf{S}^{-1}\mathbf{X}\mathbf{K}^{-1}\mathbf{S}^{-1}\mathbf{X})\mathbf{A}^T & 0\\ \mathbf{K}^{-1}\mathbf{S}^{-1}\mathbf{X}\mathbf{A}^T & I\end{bmatrix}\begin{bmatrix}d_y\\ d_{yu}\end{bmatrix} = \begin{bmatrix}-r_p - \mathbf{A}\mathbf{S}^{-1}(\mathbf{X}r_d - r_c + \mathbf{X}\mathbf{K}^{-1}r_q)\\ \mathbf{K}^{-1}r_q\end{bmatrix}$$

Let $\mathbf{D} := \mathbf{S}^{-1}\mathbf{X} - \mathbf{S}^{-1}\mathbf{X}\mathbf{K}^{-1}\mathbf{S}^{-1}\mathbf{X} = \mathbf{S}^{-1}\mathbf{X}(I - \mathbf{K}^{-1}\mathbf{S}^{-1}\mathbf{X})$ and $\mathbf{L} := \mathbf{A}\mathbf{D}\mathbf{A}^T$ and $r_s := -r_p - \mathbf{A}\mathbf{S}^{-1}(\mathbf{X}r_d - r_c + \mathbf{X}\mathbf{K}^{-1}r_q)$:

$$\begin{bmatrix}\mathbf{L} & 0\\ \mathbf{K}^{-1}\mathbf{S}^{-1}\mathbf{X}\mathbf{A}^T & I\end{bmatrix}\begin{bmatrix}d_y\\ d_{yu}\end{bmatrix} = \begin{bmatrix}r_s\\ \mathbf{K}^{-1}r_q\end{bmatrix}$$

Pretending that $\mathbf{L}$ is invertible:

$$\begin{bmatrix}I & 0\\ \mathbf{K}^{-1}\mathbf{S}^{-1}\mathbf{X}\mathbf{A}^T & I\end{bmatrix}\begin{bmatrix}d_y\\ d_{yu}\end{bmatrix} = \begin{bmatrix}\mathbf{L}^{-1}r_s\\ \mathbf{K}^{-1}r_q\end{bmatrix}$$

Eliminating d_y,

$$\delta_y = \begin{bmatrix}I & 0\\ 0 & I\end{bmatrix}\begin{bmatrix}d_y\\ d_{yu}\end{bmatrix} = \begin{bmatrix}\mathbf{L}^{-1}r_s\\ \mathbf{K}^{-1}(r_q - \mathbf{S}^{-1}\mathbf{X}\mathbf{A}^T d_y)\end{bmatrix}$$

And we can apply the remaining formula from `No KKT`:

$$\delta_x = -S^{-1}(r_c - X(r_d + A^T d - y))$$
$$\delta_s = -X^{-1}(r_c + S d_x)$$

2.ii Mehrotra's Predictor Corrector Method

Ref: Numerical Optimization (Alg. 14.3)

We reuse the same solver as `Long Step Path Following Method`, since the left hand side is the same for all systems that we have to solve, and we can simply provide a different right hand side.

3 Phase 1

We want to start the actual optimisation somewhere close to the central path, which is quite difficult to construct. So, we solved the same problem with zero cost first, to get close to the analytic centre of the feasible region.

But it turns out, even Phase-1 doesn't work well if we do not start from a feasible flow, which again is difficult to construct. So, we simply added a *star* node with 0 demand, and connected all the other nodes to it with high cost and capacity. Now it is trivial to construct a feasible flow that only routes through this *star* node.

4 Integral Flow

Ref: Flow Rounding (Sec. 4)

For the problems that converge near the optimal, we can *round* them to an integral flow by cancelling the fractional flows in $\mathcal{O}(m \log n)$. This was implemented using *link-cut trees*, although I did not verify that the implementation indeed has the claimed time-complexity.

Currently considering: Instead of rounding the flow at the end, it might be possible to round at every iteration of Phase-2 if the primal infeasibility ($\|r_p\|_\infty$) is low enough. I have not analysed the implication of this for the other termination conditions of the IPM.

5 Results

Both of the methods described above were implemented in Julia. `Mehrotra's method` was used to produce the following results. This public GitHub repository contains all relevant codes, runtime logs, and plots. We use some of them here to make a few observations.

- In all cases, Phase-1 converged relatively quickly (e.g., within 20 iterations, even for the largest problem).
- In all cases, Phase-2 started with low surrogate duality gap (μ), low primal infeasibility ($\|r_p\|_\infty$) and high dual infeasibility ($\|r_d\|_\infty$).
- For the first half of Phase-2, μ *increased*, while $\|r_d\|_\infty$ stayed high. Once μ roughly matched $\|r_d\|_\infty$, only then both started to decrease together. For the converging problems, $\|r_p\|_\infty$ stayed relatively low — but for the diverging problems, $\|r_p\|_\infty$ shot up around the same time μ matched $\|r_d\|_\infty$, and never recovered.

Figure 1 Each phase were allowed a maximum of 100 iterations. The two largest problems (`m = 166088` and `m = 309364`) diverged. Until then all the problems (up to `m = 87590`) converged with roughly logarithmic growth of number of iterations with the problem size (i.e., `n + m`)

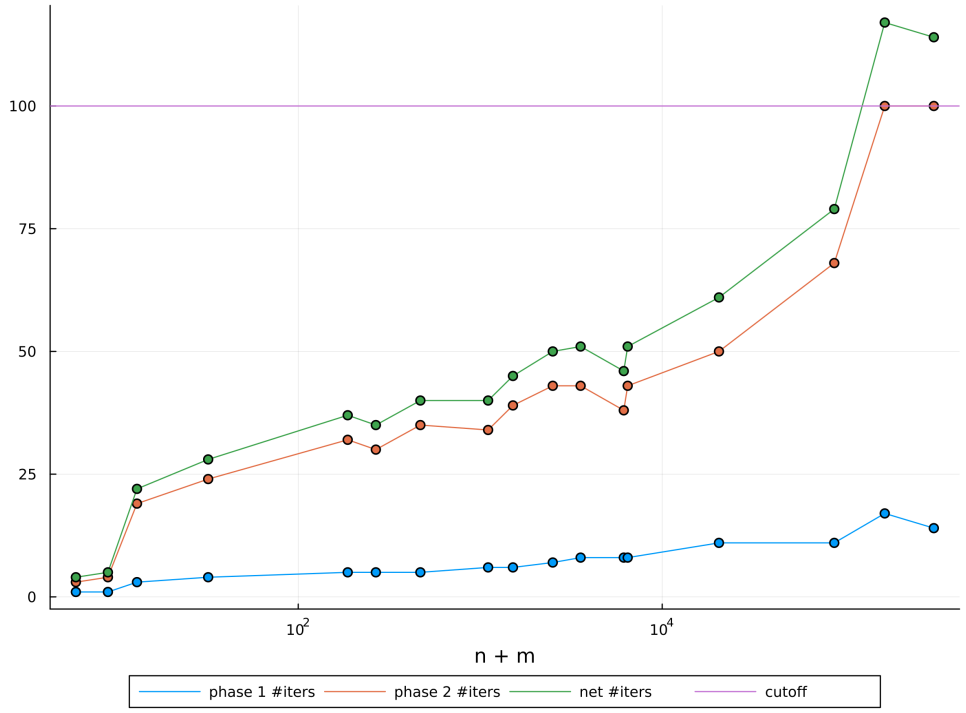
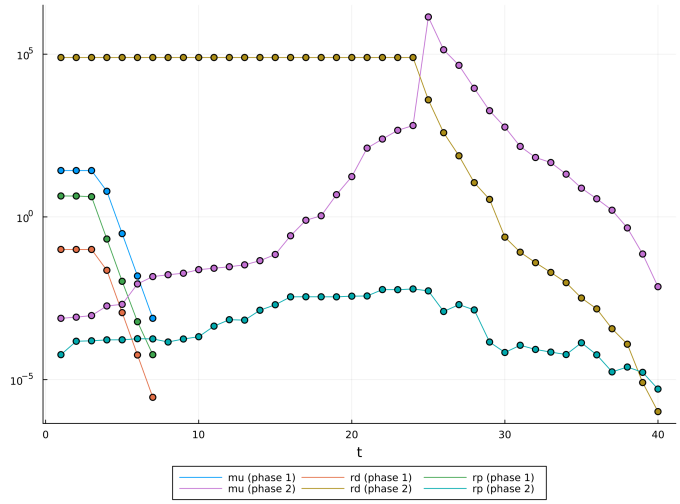
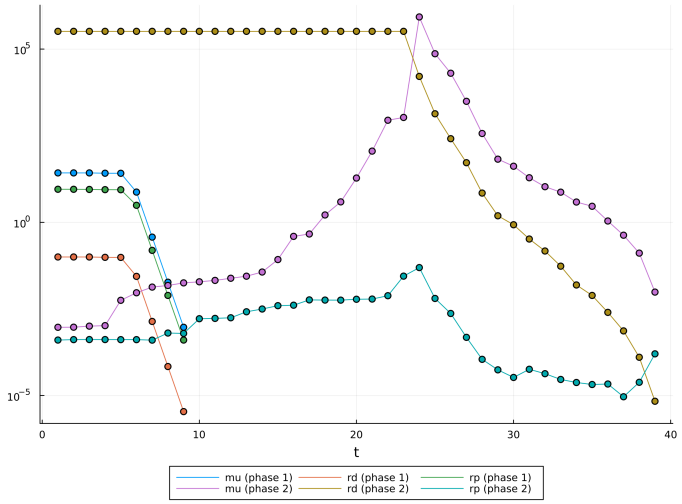


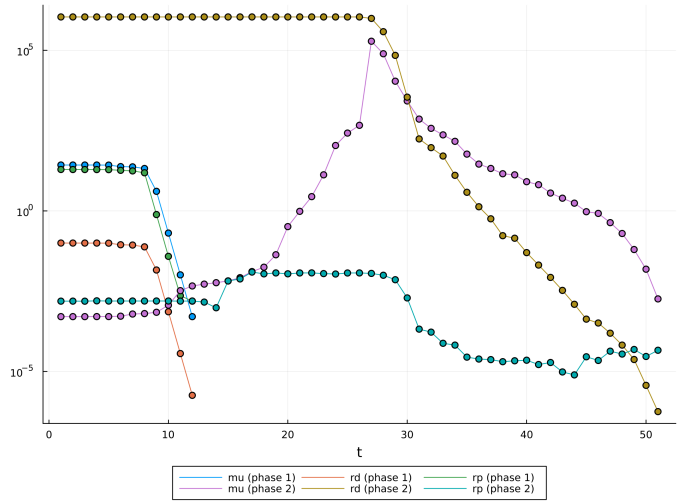
Figure 2 Progression of surrogate duality gap ($\mu = \frac{s^T x}{n}$), primal infeasibility ($\|r_p\|_\infty$), and dual infeasibility ($\|r_d\|_\infty$) for some example problems.



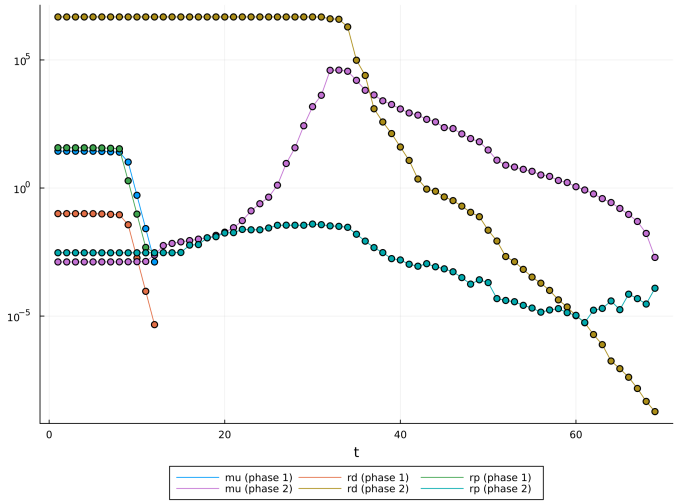
(a) `n = 50, m = 1440.`



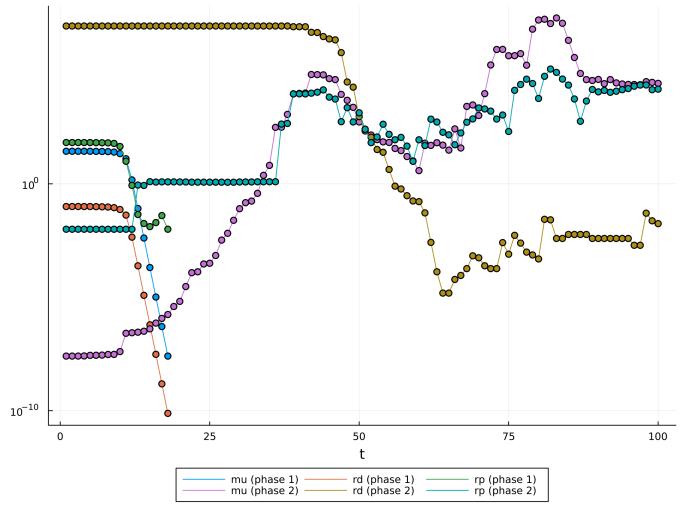
(b) `n = 100, m = 6006.`



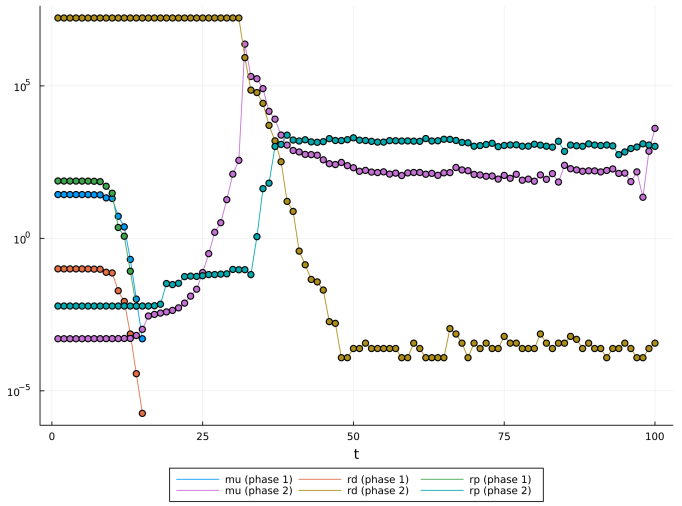
(c) `n = 300, m = 20072.`



(d) `n = 500, m = 87590.`



(e) `n = 700, m = 166088.`



(f) `n = 1000, m = 309364.`