Trace-Constrained PSD-Matrix Completion via Frank-Wolfe

Julius Graf

April 29, 2025

Given $Y \in \mathcal{S}_n^+(\mathbb{R})$ s.t. $\operatorname{tr}(Y) = 1$, consider the problem

$$(P)\colon \text{ minimize } \quad \frac{1}{2} \sum_{(i,j)\in\Omega} (y_{i,j} - x_{i,j})^2 \qquad \text{s.t.} \quad \begin{cases} \operatorname{tr}(\mathrm{X}) = 1 \\ X \succeq 0 \end{cases} .$$

Let f be the objective function of problem (P). We have $\nabla f(X) = P_\Omega(X) - P_\Omega(Y)$ for all $X \in \mathcal{M}_n(\mathbb{R})$, where $P_\Omega\colon X \in \mathcal{S}_n(\mathbb{R}) \mapsto (x_{i,j}\delta_{(i,j)\in\Omega})_{1\leqslant i,j\leqslant n} \in \mathcal{S}_n(\mathbb{R})$ is the orthogonal projector that keeps the entries in Ω and zeros out the others (clearly $P_\Omega^2 = P_\Omega$ and $P_\Omega(A) \bullet B = A \bullet P_\Omega(B)$ for all $A, B \in \mathcal{S}_n(\mathbb{R})$). Let us solve this problem using the Frank-Wolfe algorithm, using the step size $\alpha_t = 2/(t+2)$ for $t \geqslant 0$. Let $\mathcal{D} = \{X \in \mathcal{M}_n(\mathbb{R}) : X \succeq 0 \text{ and } \operatorname{tr}(X) = 1\}$.

Algorithm 1 Frank-Wolfe method to minimize $f(X)$ over $\mathcal{D}$

1: Initialize $X^0 \in \mathcal{D}$ and $t \leftarrow 0$
2: **for** $t = 0, 1, \ldots, T$ **do**
3: Compute $\nabla f(X_t) = P_\Omega(X_t) - P_\Omega(Y)$
4: Solve linear optimization problem $\tilde{X}_t \leftarrow \arg\min_{X\in\mathcal{D}}\{f(X_t) + \nabla f(X_t) \bullet (X - X_t)\}$
5: Set $\alpha_t = 2/(t+2)$
6: Update $X_{t+1} \leftarrow (1-\alpha_t)X_t + \alpha_t \tilde{X}_t$
7: **end for**

Algorithm 1 requires to solve $\min_{X\in\mathcal{D}} \nabla f(X_t) \bullet X$. To solve such a problem, we consider a more general optimization problem. Let $C \in \mathcal{S}_n(\mathbb{R})$. Consider the optimization problem

$$P(C)\colon \text{ minimize } \quad C \bullet X \qquad \text{s.t.} \quad \begin{cases} \operatorname{tr}(\mathrm{X}) = 1 \\ X \succeq 0 \end{cases} .$$

Note that we can write the problem with $C \in \mathcal{S}_n(\mathbb{R})$ because replacing $\nabla f(X_t)$ by its symmetric part does not modify the objective value. Let $\underline{\lambda}(C)$ be the smallest eigenvalue of C and $\underline{u}$ be the associated normalized eigenvector. Let us show that $X^* := \underline{u}\underline{u}^\top$ is optimal for $P(C)$. First, notice that X^* is feasible for $P(C)$ because $\operatorname{tr}(X^*) = \operatorname{tr}(\underline{u}\underline{u}^\top) = \operatorname{tr}(\underline{u}^\top \underline{u}) = \underline{u}^\top \underline{u} = 1$ and $z^\top X^* z = z^\top \underline{u}\underline{u}^\top z = (\underline{u}^\top z)^2 \geqslant 0$ for all $z \in \mathbb{R}^n$, such that $X^* \succeq 0$. Moreover, one computes that:

$$\begin{aligned} C \bullet X^* &= \operatorname{tr}(X^\top C) \\ &= \operatorname{tr}(\underline{u}^\top C \underline{u}) \\ &= \underline{\lambda}(C)\|\underline{u}\|^2 \\ &= \underline{\lambda}(C). \end{aligned}$$

Let $X \succeq 0$ such that $\operatorname{tr}(X) = 1$. By the spectral theorem, there exists $P \in \mathcal{O}_n(\mathbb{R})$ and $D \in \mathcal{M}_n(\mathbb{R})$ such that $C = PDP^\top$ and $D = \operatorname{diag}((\lambda_i))_{1 \leqslant i \leqslant n}$ for some $(\lambda_i)_{1 \leqslant i \leqslant n} \in \mathbb{R}^n$. With this notation, we denote $\underline{\lambda}(C) = \lambda_1$ and $\underline{u} = Pe_1$, where $e_1 = (\delta_{1,i})_{1 \leqslant i \leqslant n}$. More generally, denote $e_j = (\delta_{i,j})_{1 \leqslant j \leqslant n}$ for all $j \in [\![1, n]\!]$. We now write X in the eigenbasis of C, by considering $\tilde{X} = P^\top X P$. Notice that $\tilde{X} \succeq 0$ and $\operatorname{tr}(\tilde{X}) = 1$ by cyclic property of the trace. Furthermore, notice that $\tilde{x}_{i,i} = e_i^\top \tilde{X} e_i \geqslant 0$ for all $i \in [\![1, n]\!]$. Then:

$$\begin{aligned} C \bullet X &= \operatorname{tr}(CX) \\ &= \operatorname{tr}(PDP^\top X) \\ &= \operatorname{tr}(D\tilde{X}) \\ &= \sum_{i=1}^{n} \lambda_i \tilde{x}_{i,i} \\ &\geqslant \underline{\lambda}(C) \operatorname{tr}(\tilde{X}) \\ &= \underline{\lambda}(C) \\ &= C \bullet X^* \end{aligned}$$

Hence, $C \bullet X \geqslant C \bullet X^*$. Thus, X^* is optimal for $P(C)$.

Theorem 1. *Let $M \in \mathcal{S}_q(\mathbb{R})$ with eigenvalues $\lambda_1 < \lambda_2 \leqslant \cdots \leqslant \lambda_q$ and let $(v_i)_{1 \leqslant i \leqslant q}$ be an orthonormal eigenbasis. Set*

$$\sigma = \min_{1 \leqslant i \leqslant q} \left(m_{i,i} - \sum_{j \neq i}^{q} |m_{i,j}| \right) - \varepsilon < \lambda_1$$

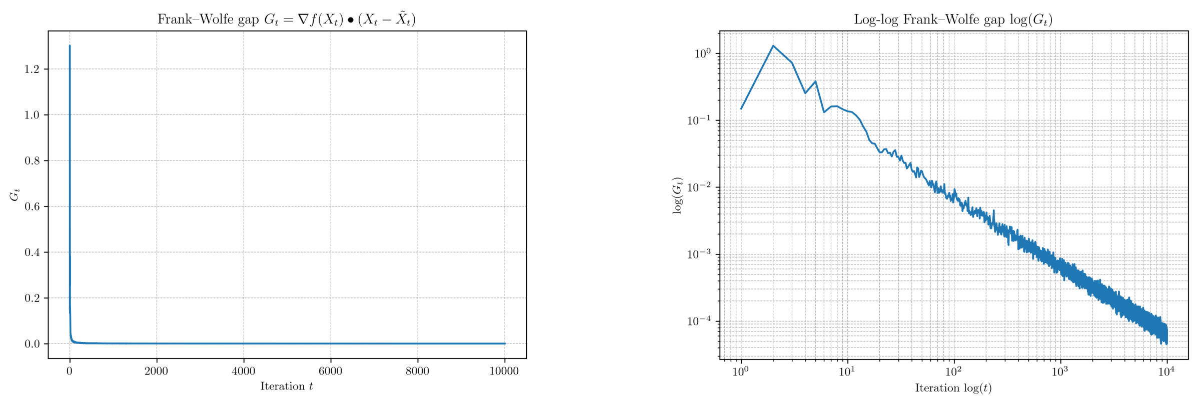
for some small $\varepsilon > 0$ (s.t. $M - \sigma I \succ 0$) and let $x_0 = \mu v_1 + f$ with $\mu \neq 0$ and $f \in \operatorname{im}(M - \lambda_1 I)$. Define the sequence

$$\forall n \geqslant 0, \quad x_{n+1} = \frac{(M - \sigma I)^{-1} x_n}{\|(M - \sigma I)^{-1} x_n\|_2}.$$

Then $x_n \xrightarrow[n \to +\infty]{} \pm v_1$ and $x_n^\top M x_n \underset{n \to +\infty}{\uparrow} \lambda_1$.

Thus, for our problem, $P(\nabla f(X_t))$, we simply need to compute $\underline{\lambda}(\nabla f(X_t))$, find an eigenvector $\underline{u}(\nabla f(X_t))$ at each iteration and then update $X_{t+1} = (1-\alpha_t)X_t + \alpha_t \underline{u}(\nabla f(X_t))\underline{u}(\nabla f(X_t))^\top$.[1]

We run the algorithm using the provided data file for the observed entries of a 50x50 matrix Y, given as a triplet list `Y_262b_Spring2022.csv`, for a total of 10,000 iterations.



(a) Frank-Wolfe gap $G_t = \nabla f(X_t) \bullet (X_t - \tilde{X}_t)$

(b) Log-log Frank-Wolfe gap $\log(G_t)$

Figure 1: Convergence of the Frank-Wolfe algorithm on the PSD matrix-completion problem

[1] Using the notation of theorem 1, we compute $\lambda_1 = \underline{\lambda}(\nabla f(X_t))$ and $v_1 = \underline{u}(\nabla f(X_t))$, to unify notations.

Appendix

```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import matplotlib as mpl

mpl.rcParams["text.usetex"] = True
mpl.rcParams["text.latex.preamble"] = r"\usepackage{amsmath}\usepackage{amssymb}"
mpl.rcParams["font.family"] = "serif"

CSV_FILE   = "Y_262b_Spring2025.csv"   # input data file
N          = 50                        # matrix dimension (50 x 50)
N_ITERS    = 10_000                    # assignment asks for 10 000 iterations
SAVE_PLOTS = True                      # set False to just display

def step_size(t: int) -> float:
    return 2.0 / (t + 2)

# ---------- load triplet data ----------
triplets = pd.read_csv(CSV_FILE).to_numpy(float)
rows = triplets[:, 0].astype(int) - 1   # 0-based
cols = triplets[:, 1].astype(int) - 1
vals = triplets[:, 2]

# ---------- build mask and observed-entry matrix ----------
mask = np.zeros((N, N), dtype=bool)
Yobs = np.zeros((N, N))
for r, c, v in zip(rows, cols, vals):
    if not (0 <= r < N and 0 <= c < N):
        raise ValueError(f"Index ({r+1},{c+1}) out of range for {N}x{N} matrix.")
    mask[r, c] = mask[c, r] = True       # symmetry
    Yobs[r, c] = Yobs[c, r] = v

def P_Omega(M: np.ndarray) -> np.ndarray:
    R = np.zeros_like(M)
    R[mask] = M[mask]
    return R

# ---------- initialise ----------
X = np.eye(N) / N            # trace = 1, PSD
gaps = np.empty(N_ITERS)

for t in range(N_ITERS):
    grad = P_Omega(X) - Yobs
    eigvals, eigvecs = np.linalg.eigh(grad)
    u_min = eigvecs[:, np.argmin(eigvals)]
    X_tilde = np.outer(u_min, u_min)

    gaps[t] = np.sum(grad * (X - X_tilde))

    alpha = step_size(t)
    X = (1 - alpha) * X + alpha * X_tilde

    if (t + 1) % 1_000 == 0:
        print(f"iter {t+1:>5d}  gap = {gaps[t]:.3e}")

# ---------- plots ----------
iters = np.arange(1, N_ITERS + 1)

fig1, ax1 = plt.subplots()
ax1.plot(iters, gaps)
```

```
ax1.set_xlabel("Iteration $t$")
ax1.set_ylabel("$G_t$")
ax1.set_title("Frank-Wolfe gap $G_t = \\nabla f(X_t) \\bullet (X_t - \\tilde{X}_t)$")
ax1.grid(True, ls="--", lw=0.5)
fig1.tight_layout()

fig2, ax2 = plt.subplots()
ax2.loglog(iters, gaps)
ax2.set_xlabel("Iteration $\log(t)$")
ax2.set_ylabel("$\log(G_t)$")
ax2.set_title("Log-log Frank-Wolfe gap $\log(G_t)$")
ax2.grid(True, ls="--", lw=0.5, which="both")
fig2.tight_layout()

if SAVE_PLOTS:
    fig1.savefig("gap_linear.png", dpi=300)
    fig2.savefig("gap_loglog.png", dpi=300)
    print("Plots saved as gap_linear.png and gap_loglog.png")
else:
    plt.show()
```

Listing 1: Frank-Wolfe solver for PSD-matrix completion with unit trace