Advanced Graph Algorithms and Optimization Spring 2024

Spectral Graph Theory

R. Kyng & M. Probst *Problem Set 4 — Monday, March 11*

These exercises will not count toward your grade, but you are encouraged to solve them all. This exercise sheet contains exercises relating to lectures in Week 4. We encourage you to start the exercises early so you have time to get through everything.

To get feedback, you must hand in your solutions by 23:59 on March 21. Both hand-written and LaTeX solutions are acceptable, but we will only attempt to read legible text.

Exercise 1

The bound obtained in Cheeger's inequality is indeed tight. Prove that:

1. Let G be the graph consisting of two vertices connected by a single edge of unit weight. Prove that $\phi(G) = \lambda_2(\boldsymbol{N})/2$ and therefore that the lower bound of Cheeger's inequality is tight.
2. To show that the line graph proves that the upper bound of Cheeger's Inequality is asymptotically tight (i.e. up to constant factors).

Solution

1. To prove the tightness of lower bound, consider the graph G of two vertices connected by a single edge. It is easy to compute that $\phi(G) = 1$, where only one partition of vertices is possible. We now compute $\lambda_2(\boldsymbol{N})$.

$$\boldsymbol{D} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad \boldsymbol{A} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \quad \boldsymbol{L} = \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}.$$

   Since $\mathrm{Tr}(\boldsymbol{L}) = \lambda_1(\boldsymbol{L}) + \lambda_2(\boldsymbol{L}) = 2$, we have $\lambda_2(\boldsymbol{L}) = 2 - 0 = 2$.

$$\lambda_2(\boldsymbol{N}) = \lambda_2(\boldsymbol{D}^{-\frac{1}{2}}\boldsymbol{L}\boldsymbol{D}^{-\frac{1}{2}}) = \lambda_2(\boldsymbol{L}) = 2.$$

   Therefore, we have shown that $\phi(G) = \lambda_2(\boldsymbol{N})/2 = 1$ and therefore, the lower bound of Cheeger's inequality is tight.

2. To prove the tightness of upper bound asymptotically, consider the path graph P_n. It is known that $\phi(P_n) = \Theta\left(\frac{1}{n}\right)$ from Exercise 1. Now, we compute $\lambda_2(\boldsymbol{N})$. We use the test vector $\boldsymbol{x}$ for the second eigenvalue of $\boldsymbol{N}$, where $\boldsymbol{x}(i) = (n+1) - 2i, i \in [n]$. This vector satisfies

$\boldsymbol{x} \perp \boldsymbol{d}$ since $\boldsymbol{d}(i) = 2$ for all i except $\boldsymbol{d}(1) = \boldsymbol{d}(n) = 1$. By Courant-Fischer, we have

$$\begin{aligned}\lambda_2(\boldsymbol{N}) = \min_{\boldsymbol{z}\perp\boldsymbol{d},\boldsymbol{z}\neq 0} \frac{\boldsymbol{z}^\top \boldsymbol{L}\boldsymbol{z}}{\boldsymbol{z}^\top \boldsymbol{D}\boldsymbol{z}} \leq \frac{\boldsymbol{x}^\top \boldsymbol{L}\boldsymbol{x}}{\boldsymbol{x}^\top \boldsymbol{D}\boldsymbol{x}} &= \frac{\sum_{i\in[n-1]}(\boldsymbol{x}(i) - \boldsymbol{x}(i+1))^2}{\sum_{i\in[n]} \boldsymbol{d}(i)\boldsymbol{x}(i)^2} \\ &= \frac{\sum_{i\in[n-1]} 2^2}{2\left(\sum_{i\in[n]}(n+1-2i)^2 - (n-1)^2\right)} \\ &= \frac{4(n-1)}{2[(n+1)n(n-1)/3 - (n-1)^2]} \\ &= \Theta\left(\frac{1}{n^2}\right).\end{aligned}$$

Thus, we can prove that the upper bound is asympototically tight by obtaining

$$\Theta\left(\frac{1}{n}\right) = \phi(P_n) \leq \sqrt{2\lambda_2(\boldsymbol{N})} \leq \sqrt{2\Theta\left(\frac{1}{n^2}\right)} = \Theta\left(\frac{1}{n}\right).$$

Exercise 2

Sparse Expanders: In random graph theory, the graph over n vertices where each edge between two endpoints is present independently with probability p is denoted $G(n,p)$.

Show that for $p = \Omega(\log n/n)$, that $G(n,p)$ is a $\Omega(1)$-expander with high probability (it is up to you to fix large constants). Take the following steps:

1. Prove that with high probability, $\boldsymbol{d}(u) = \Theta(pn)$ for all vertices $u \in V(G(n,p))$.
2. For each set S of $k \leq n/2$ vertices, argue that

$$\mathbb{P}[|E(S, V\setminus S)| = \Theta(kpn)] > 1 - n^{-c\cdot k}$$

   for any large constant $c > 0$.
3. Observe that there are at most $\binom{n}{k}$ sets of vertices S of size k. Conclude that $G(n,p)$ is with high probability a $\Omega(1)$-expander.

Solution

1. For any vertex in $u \in V$,

$$X_i := \begin{cases} 1 & \text{if } u \text{ is connected to } v_i \in V \setminus u,\ i \in [n-1] \\ 0 & \text{o.w.} \end{cases}$$

   is a Bernoulli random variable with $\mathbb{P}(X_i = 1) = p = \Omega(\log n/n)$. Also, if $X = \sum_{i\in[n-1]} X_i = \boldsymbol{d}(u)$ is the random variable denoting the degree of u, then we have

$$\mathbb{E}[X] = (n-1)p = \Theta(np).$$

By Chernoff bound, choosing $p = \Theta\left(\frac{\log n}{\epsilon^2 n}\right)$ with $\epsilon \in (0, 1)$ being a constant, we have

$$\begin{aligned}\mathbb{P}\left[|X - \mathbb{E}[X]| \geq \epsilon\mathbb{E}[X]\right] &\leq 2e^{-\frac{1}{3}\epsilon^2\mathbb{E}[X]} \\ &= 2e^{-\frac{1}{3}\epsilon^2\Theta(np)} \\ &= 2e^{-\Theta(\log n)} = 2n^{-c}\end{aligned}$$

for some large constant $c > 0$. Therefore, we have proved that $\boldsymbol{d}(u) \in [(1-\epsilon)\mathbb{E}[X], (1+\epsilon)\mathbb{E}[X]] = \Theta(pn)$ with probability larger than $1 - 2n^{-c}$. Then, we use union bound over all n vertices, and we have that for all vertices $u \in V$, $\boldsymbol{d}(u) = \Theta(pn)$ with probability larger than $1 - 2n^{1-c} := 1 - 2n^{-c'}$, for some large constant $c' > 0$.

2. For any set S with $k = |S| \leq n/2$,

$$Y_i := \begin{cases} 1 & \text{if edge } i \text{ is sample, which has exactly one endpoint in } S \\ 0 & \text{o.w.} \end{cases}$$

is a Bernoulli random variable with $\mathbb{P}(Y_i = 1) = p = \Omega(\log n/n)$. Also, if $Y = \sum_{i\in[k(n-k)]} Y_i = E(S, V \setminus S)|$ is the random variable denoting the number of cut edges S, then we have

$$\mathbb{E}[Y] = k(n-k)p = \Theta(kpn).$$

By Chernoff bound, also choosing $p = \Theta\left(\frac{\log n}{\epsilon^2 n}\right)$ with $\epsilon \in (0, 1)$ being a constant, we have

$$\begin{aligned}\mathbb{P}\left[|Y - \mathbb{E}[Y]| \geq \epsilon\mathbb{E}[Y]\right] &\leq 2e^{-\frac{1}{3}\epsilon^2\mathbb{E}[Y]} \\ &= 2e^{-\frac{1}{3}\epsilon^2\Theta(kpn)} \\ &= 2e^{-\Theta(k\log n)} = 2n^{-ck}\end{aligned}$$

for some large constant $c > 0$. Therefore, we have proved that for some set $S \subset V$, $|E(S, V \setminus S)| \in [(1-\epsilon)\mathbb{E}[Y], (1+\epsilon)\mathbb{E}[Y]] = \Theta(kpn)$ with probability larger than $1 - 2n^{-ck}$.

3. For any $k \leq n/2$, there are at most $\binom{n}{k} = \Theta(n^k)$ set of vertices S. We first use union bound over these sets with size k. Then for all S of size k, $|E(S, V \setminus S)| = \Theta(kpn)$ with probability larger than $1 - 2n^{(1-c)k} := 1 - 2n^{-c'k}$. Then, using a union bound again over all possible $1 \leq k \leq n/2$ yields

$$\sum_{k=1}^{n/2} 2n^{-c'k} \leq \frac{n}{2} 2n^{-c'} \leq O(n^{1-c'}).$$

Therefore, we can conclude that for all set S with all $k \leq n/2$, $|E(S, V \setminus S)| = \Theta(kpn)$ with probability larger than $1 - n^{(1-c')} := 1 - n^{-c''}$, for some large constant $c'' > 0$. Hence, with high probability, we have

$$\phi(G(n,p)) = \min_{\emptyset \subset S \subset V, |S| \leq n/2} \frac{|E(S, V \setminus S)|}{vol(S)} \leq \frac{\Theta(kpn)}{k\Theta(pn)} = \Theta(1),$$

thus $G(n, p)$ is a $\Omega(1)$-expander with high probability.

Exercise 3

Let $G = (V, E)$ be a connected, undirected graph. In this problem, you will show that there is an algorithm that computes a ϕ-expander decomposition $X_1, X_2, \ldots, X_k$ for G of quality $q = O(\phi^{-1/2} \cdot \log n)$ in time $O(m \log^c n)$ for some constant c. We let $\boldsymbol{N}$ denote the normalized Laplacian in this exercise, defined by $\boldsymbol{N} = \boldsymbol{D}^{-1/2}\boldsymbol{L}\boldsymbol{D}^{-1/2}$.

Assume that you are given an algorithm $\textsc{CertifyOrCut}(G, \phi)$ that given a graph G and a parameter ϕ either:

- *Certifies* that G is a ϕ-expander, or
- Presents a *cut* S such that $\phi(S) = O(\sqrt{\phi})$.

The algorithm $\textsc{CertifyOrCut}(G, \phi)$ runs in time $O(m \log^{c'} n)$ for $c' > 0$.

1. Show that there is an algorithm that uses $\textsc{CertifyOrCut}(G, \phi)$ and computes a ϕ-expander decomposition of quality $O(\phi^{-1/2} \cdot \log n)$ in time $O(mn \cdot \log^{c'} n)$.
2. Show that in $O(mn \cdot \log^{c'} n)$ time, you can implement a procedure $\textsc{CertifyOrLargeCut}(G, \phi)$ that outputs a set S (possibly empty) with $\phi(S) = O(\sqrt{\phi})$ such that either
   - $G[V \setminus S]$ is a ϕ-expander and $vol_G(V \setminus S) \geq \frac{1}{3}m$, or
   - $\min\{vol_G(S), vol_G(V \setminus S)\} \geq \frac{1}{3}m$.

   Comment: You may use the following statement.

   Claim: Given a set $S \subseteq V$ of conductance $\phi_G(S) \leq \phi$ and a set $S' \subseteq V \setminus S$ in $G[V \setminus S]$ with conductance $\phi_{G[V \setminus S]}(S') \leq \phi$, you have that $\phi_G(S \cup S') \leq \phi$.
3. Using a local version of random walks and heavy randomization, it turns out that $\textsc{CertifyOrLargeCut}(G, \phi)$ can be implemented in $O(m \log^{c''} n)$ time[1]. Show that this implies an $O(m \log^{c''+1} n)$ time algorithm to compute a ϕ-expander decomposition.

Solution.

1. We repeatedly apply $\textsc{CertifyOrCut}(G, \phi)$ to different induced subgraphs, containing fewer and fewer vertices. Our algorithm can be seen as a sequence of l steps, where we obtain a partition $X_{i,1}, X_{i,1}, ..., X_{i,k_i}$ of X for each step $i \leq l$. For each such set of vertices we also keep a Boolean variable, that marks the set either as $finished$, or not.

   Initially, we set $X_{0,1} = X$, a completely trivial decomposition, and mark it as not finished. Then, during step i of the algorithm, we apply $\textsc{CertifyOrCut}(G[X_{i-1,j}], \phi)$ for each $1 \leq j \leq k_{i-1}$ such that $X_{i-1,j}$ that is not marked as finished. If the algorithm returns a cut $S, X_{i-1,j} \setminus S$, we add S and $X_{i-1,j} \setminus S$ as two separate sets to the partition of step i and mark them as not finished, otherwise we just add $X_{i-1,j} \setminus S$ and mark it as finished. Finally we

[1] Note that this is slightly idealized for ease of presentation.

include all the finished sets from the previous step in the partition of step i. The algorithm terminates once all sets of a step are finished.

We start by showing that this algorithm meets the runtime requirements. Let $G[X_{i-1,j}] = (V_{i-1,j}, E_{i-1,j})$ and consider a specific step i. We call $|V_{i-1,j}| = n_j$ and $E_{i-1,j} = m_j$. Then the computational cost of this step is upper bounded by

$$\sum_{j=1}^{k_{i-1}} O(m_j \log^{c'} n_j) \leq O(m \log^{c'} n).$$

Since the size of each unfinished set decreases by at least 1 in each step, the total number of steps is bounded by n. We conclude the total runtime of

$$O(nm \log^{c'} n)$$

as desired.

It is clear that the resulting partition of this algorithm will be a ϕ-expander decomposition, since all the sets in the final partition are marked as finished. It remains to be shown that we have at most $O(\log n\sqrt{\phi}m)$ edges going across components. We will show this by induction on the number of edges in a graph. Our induction hypothesis goes as follows: Our algorithm splits a graph with m' edges and n' vertices into components, such that at most

$$c\log(m'+1)\sqrt{\phi}m'$$

edges go across. For singleton vertices this clearly holds. Consider some other value m'. Either our algorithm initially splits the graph into two components, or we are done. If it splits it in two components, with m'_1 and m'_2 edges respectively, we have that the number of edges that go across components is bound by

$$c\log(m'_1+1)\sqrt{\phi}m'_1 + c\log(m'_2+1)\sqrt{\phi}m'_2 + cm'_2\sqrt{\phi} \leq c\log(m'+1)\sqrt{\phi}m'$$

for $m'_2 \leq m'_1$since G is connected. The result follows by induction and $m \leq n^2$.

2. We start by proving the claim.

Claim: Prove that given a set $S \subseteq V$ of conductance $\phi_G(S) \leq \phi$ and a set $S' \subseteq V \setminus S$ in $G[V \setminus S]$ with conductance $\phi_{G[V\setminus S]}(S') \leq \phi$, you have that $\phi_G(S \cup S') \leq \phi$ if $vol(S \cup S') \leq vol(V \setminus S \cup S')$.

We let S and S' denote the sides with smaller volume and let $vol(S \cup S') \leq vol(V \setminus S \cup S')$. We have

$$\begin{aligned}|E(S, V \setminus S)| &\leq \phi \min\{vol(S), vol(V \setminus S)\}\\ &\leq \phi vol(S)\end{aligned}$$

and

$$\begin{aligned}|E_{G[V\setminus S]}(S', V \setminus (S \cup S'))| &\leq \phi \min\{vol_{G[V\setminus S]}(S'), vol_{G[V\setminus S]}(V \setminus (S \cup S'))\}\\ &\leq \phi vol_{G[V\setminus S]}(S').\end{aligned}$$

Then we calculate

$$\begin{aligned}|E(S \cup S', V \setminus (S \cup S'))| &\leq |E(S, V \setminus S)| + |E_{G[V \setminus S]}(S', V \setminus (S \cup S'))| \\ &\leq \phi(vol(S) + vol_{G[V \setminus S]}(S')) \\ &\leq \phi \min\{vol(S \cup S'), vol(V \setminus (S \cup S'))\}.\end{aligned}$$

Now that we know this our algorithm works in a very simple way. We call $\textsc{CertifyOrCut}(G, \phi)$ on the full graph initially. If it certifies G being a ϕ-expander, we are done. Otherwise it returns two sets S and $V \setminus S$, where we assume S is the one with smaller volume. If S has volume at least $\tau = 1/3m$ we are done. Otherwise, we recurse on the graph $G[V \setminus S]$ with $\tau = \tau - |S|$. Each recursive call reduces the size of the graph by at least one vertex, which yields the desired run time. When we terminate, either the total volume of the set of discarded vertices is $1/3m$, or the remaining vertices form a ϕ-expander with volume $1/3m$ or more. Consider the sets $S_1, S_2, ...$ of vertices removed in each recursive iteration. We inductively apply the hint on these and get that the total size of the cut does not exceed $O(\phi)$. We need to be a bit careful the last time we apply the hint, since there the condition $vol(S \cup S') \leq vol(V \setminus S \cup S')$ might not apply. However, we can replace it with $\frac{1}{2}vol(S \cup S') \leq vol(V \setminus S \cup S')$ and pay with an increased in conductance by a factor of 2.

3. (BONUS) We use the same algorithm as for the first question, but replace $\textsc{CertifyOrCut}(G, \phi)$ with $\textsc{CertifyOrLargeCut}(G, \phi)$ throughout. Intuitively this should make the recursive tree have limited depth, since $\textsc{CertifyOrLargeCut}(G, \phi)$ enforces a level of balance. To formally show this is the case, we introduce a potential function. Consider a partition $V = \bigcup_{i=1}^{k} X_i$, where all X_i are pairwise disjoint. We define the potential as

$$\Psi = \sum_{i=1}^{k} \log(|E|/|E(G[X_i])|)|E(G[X_i])| + \log |E|\gamma$$

where γ denotes the number of edges that go across two sets in the partition. Initially γ is 0 and we always have $\Psi \leq 3\log(n)m$. We intend to show, that each call to $\textsc{CertifyOrLargeCut}(G, \phi)$ increases the potential substantially. Consider a call to $\textsc{CertifyOrLargeCut}(G, \phi)$. In such a call we take an X_i and split it into two sets X_l and X_p, or we stop the recursion on this branch. If we split it into two sets, we increase the potential function by

$$\alpha = \log(|E|/m_l)m_l + \log(|E|/m_p)|)m_p - \log(|E|/m_i)m_i + \log |E|\gamma'$$

where m_j is shorthand for $|E(G[X_j])|$ and γ' denotes the number of edges going across the cut. We have $\frac{2}{3}m_i \geq m_l \geq \frac{1}{3}m_i$ and $\frac{2}{3}m_i \geq m_p \geq 1/3m_i$ by the guarantees of $\textsc{CertifyOrLargeCut}(G, \phi)$. Thus we have

$$\begin{aligned}\alpha &\geq \log(|E|/m_i)(m_l + m_p) - \log(2/3)(m_l + m_p) - \log(|E|/m_i)m_i + \log |E|\gamma' \\ &\geq \log(|E|/m_i)(m_l + m_p) - (2/3)\log(2/3)m_i - \log(|E|/m_i)m_i \\ &\geq (2/3)\log(2/3)m_i.\end{aligned}$$

Therefore the potential increases by a linear factor in m_i, whenever we call $\textsc{CertifyOrLargeCut}(G, \phi)$ for a graph with m_i edges. The upper bound on the potential function lets us conclude the resulting runtime.

Exercise 4

For a matrix $\boldsymbol{Z}$ to be the pseudoinverse of a symmetric matrix $\boldsymbol{M}$, you need to show that

1. $\boldsymbol{Z}^\top = \boldsymbol{Z}$.
2. $\boldsymbol{Z}\boldsymbol{v} = \boldsymbol{0}$ for $\boldsymbol{v} \in \ker(\boldsymbol{M})$.
3. $\boldsymbol{M}\boldsymbol{Z}\boldsymbol{v} = \boldsymbol{v}$ for $\boldsymbol{v} \in \ker(\boldsymbol{M})^\perp$.

Prove that if $\boldsymbol{Z}$ and $\boldsymbol{Y}$ are both the pseudo-inverse of symmetric matrix $\boldsymbol{M}$, then $\boldsymbol{Z} = \boldsymbol{Y}$, i.e. the pseudo-inverse is unique.

Solution

Assume $\boldsymbol{Z} \neq \boldsymbol{Y}$. Then there exists some test vector $\boldsymbol{v}$, such that $(\boldsymbol{Z} - \boldsymbol{Y})\boldsymbol{v} \neq \boldsymbol{0}$. Let $\boldsymbol{v} = \boldsymbol{v}_1 + \boldsymbol{v}_2$ with $\boldsymbol{v}_1 \in \ker(\boldsymbol{M})^\perp$ and $\boldsymbol{v}_2 \in \ker(\boldsymbol{M})$. But then consider

$$\boldsymbol{M}(\boldsymbol{Z} - \boldsymbol{Y})\boldsymbol{v} = \boldsymbol{M}\boldsymbol{Z}\boldsymbol{v}_1 - \boldsymbol{M}\boldsymbol{Y}\boldsymbol{v}_1 + \boldsymbol{M}\boldsymbol{Z}\boldsymbol{v}_2 - \boldsymbol{M}\boldsymbol{Y}\boldsymbol{v}_2 = \boldsymbol{v}_1 - \boldsymbol{v}_1 = \boldsymbol{0}$$

which implies $(\boldsymbol{Z} - \boldsymbol{Y})\boldsymbol{v} \in \ker(\boldsymbol{M})$. However, this cannot be since $\operatorname{im}(\boldsymbol{Z})^\perp = \operatorname{im}(\boldsymbol{Z}^T)^\perp = \ker(\boldsymbol{Z}) \subseteq \ker(\boldsymbol{M})$ and thus $\operatorname{im}(\boldsymbol{Z} - \boldsymbol{Y}) \perp \ker(\boldsymbol{M})$.

Exercise 5

Let $\boldsymbol{M} = \boldsymbol{X}\boldsymbol{Y}\boldsymbol{X}^\top$ for some $\boldsymbol{X}, \boldsymbol{Y} \in \mathbb{R}^{n\times n}$, where $\boldsymbol{X}$ is invertible and $\boldsymbol{M}$ is symmetric. Furthermore, consider the spectral decomposition of $\boldsymbol{M} = \sum_{i=1}^n \lambda_i \boldsymbol{v}_i \boldsymbol{v}_i^\top$. Then, we define $\boldsymbol{\Pi}_{\boldsymbol{M}} = \sum_{i,\lambda_i \neq 0} \boldsymbol{v}_i \boldsymbol{v}_i^\top$. $\boldsymbol{\Pi}_{\boldsymbol{M}}$ is the orthogonal projection onto the image of $\boldsymbol{M}$: It has the property that for $\boldsymbol{v} \in \operatorname{im}(\boldsymbol{M})$, $\boldsymbol{\Pi}_{\boldsymbol{M}}\boldsymbol{v} = \boldsymbol{v}$ and for $\boldsymbol{v} \in \ker(\boldsymbol{M})$, $\boldsymbol{\Pi}_{\boldsymbol{M}}\boldsymbol{v} = \boldsymbol{0}$.

Prove that

$$\boldsymbol{Z} = \boldsymbol{\Pi}_{\boldsymbol{M}}(\boldsymbol{X}^\top)^{-1}\boldsymbol{Y}^+\boldsymbol{X}^{-1}\boldsymbol{\Pi}_{\boldsymbol{M}}$$

is the pseudoinverse of $\boldsymbol{M}$.

Solution

1. Since $(\boldsymbol{v}_i\boldsymbol{v}_i^\top)^\top = \boldsymbol{v}_i\boldsymbol{v}_i^\top$, we have that $\boldsymbol{\Pi}_{\boldsymbol{M}}^\top = \boldsymbol{\Pi}_{\boldsymbol{M}}$. Furthermore, we observe that $(\boldsymbol{X}^\top)^{-1} = (\boldsymbol{X}^{-1})^\top$ and $(\boldsymbol{Y}^+)^\top = \boldsymbol{Y}^+$. Therefore, we get

$$\begin{aligned}\boldsymbol{Z}^\top &= \left(\boldsymbol{\Pi}_{\boldsymbol{M}}(\boldsymbol{X}^\top)^{-1}\boldsymbol{Y}^+\boldsymbol{X}^{-1}\boldsymbol{\Pi}_{\boldsymbol{M}}\right)^\top \\ &= \boldsymbol{\Pi}_{\boldsymbol{M}}^\top \left(\boldsymbol{X}^{-1}\right)^\top \left(\boldsymbol{Y}^+\right)^\top \left(\left(\boldsymbol{X}^\top\right)^{-1}\right)^\top \boldsymbol{\Pi}_{\boldsymbol{M}}^\top \\ &= \boldsymbol{\Pi}_{\boldsymbol{M}} \left(\boldsymbol{X}^\top\right)^{-1} \boldsymbol{Y}^+\boldsymbol{X}^{-1}\boldsymbol{\Pi}_{\boldsymbol{M}} \\ &= \boldsymbol{Z}.\end{aligned}$$

2. Consider an arbitrary vector $\boldsymbol{v} \in \ker(\boldsymbol{M})$. We know that $\boldsymbol{\Pi}_{\boldsymbol{M}} \boldsymbol{v} = \boldsymbol{0}$. Therefore, we have

$$\boldsymbol{Z}\boldsymbol{v} = \boldsymbol{\Pi}_{\boldsymbol{M}} \left(\boldsymbol{X}^{\top}\right)^{-1} \boldsymbol{Y}^{+}\boldsymbol{X}^{-1}\boldsymbol{\Pi}_{\boldsymbol{M}}\boldsymbol{v} = \boldsymbol{0}.$$

3. Let $\boldsymbol{v}$ be an arbitrary vector in $\ker(\boldsymbol{M})^{\perp}$. We want to prove that $\boldsymbol{M}\boldsymbol{Z}\boldsymbol{v} = \boldsymbol{v}$. We know that $\ker(\boldsymbol{M})^{\perp} = \operatorname{im}(\boldsymbol{M})$. This implies that $\boldsymbol{v} = \boldsymbol{M}\boldsymbol{z}$ for some vector $\boldsymbol{z}$. Therefore, it suffices to show that $\boldsymbol{M}\boldsymbol{Z}\boldsymbol{M}\boldsymbol{z} = \boldsymbol{M}\boldsymbol{z}$. We observe that $\boldsymbol{M}\boldsymbol{\Pi}_{\boldsymbol{M}} = \boldsymbol{\Pi}_{\boldsymbol{M}}\boldsymbol{M} = \boldsymbol{M}$. Thus, we have that

$$\begin{aligned}
\boldsymbol{M}\boldsymbol{Z}\boldsymbol{M}\boldsymbol{z} &= (\boldsymbol{M}\boldsymbol{\Pi}_{\boldsymbol{M}}) \left(\boldsymbol{X}^{\top}\right)^{-1} \boldsymbol{Y}^{+}\boldsymbol{X}^{-1} (\boldsymbol{\Pi}_{\boldsymbol{M}}\boldsymbol{M})\, \boldsymbol{z} \\
&= \boldsymbol{M} \left(\boldsymbol{X}^{\top}\right)^{-1} \boldsymbol{Y}^{+}\boldsymbol{X}^{-1}\boldsymbol{M}\boldsymbol{z} \\
&= \boldsymbol{X}\,\boldsymbol{Y}\boldsymbol{X}^{\top} \left(\boldsymbol{X}^{\top}\right)^{-1} \boldsymbol{Y}^{+}\boldsymbol{X}^{-1}\boldsymbol{X}\,\boldsymbol{Y}\boldsymbol{X}^{\top}\boldsymbol{z} \\
&= \boldsymbol{X}\,\boldsymbol{Y}\,\boldsymbol{Y}^{+}\boldsymbol{Y}\boldsymbol{X}^{\top}\boldsymbol{z} \\
&= \boldsymbol{X}\,\boldsymbol{Y}\boldsymbol{X}^{\top}\boldsymbol{z} \\
&= \boldsymbol{M}\boldsymbol{z}.
\end{aligned}$$