# Optimization on Manifolds: Estimating rotations from relative measurements

Thomas RENARD, Benoît MÜLLER

EPFL – MATH-512 – Project 2

May 2023

## 1 Application description and warm-up problems

**1.**

If no anchor is provided, we know the "relative position" but not the "absolute position" in the space: without noise, we could pre-compose every rotation by an arbitrary rotation and have the exact same relative rotations:

$$\forall R \in \mathrm{SO}(d),\ (R_i R)(R_j R)^\top = R_i R R^\top R_j^\top = R_i R_j^\top = H_{ij}.$$

**2.**

Only one anchor is needed. The neighborhood of a known rotation can be computed since $R_i R_j^\top = H_{ij}$ if and only if $R_i = H_{ij} R_j$. Repeating this process from the anchor to the rest of the graph allows to compute the rotations since the graph is connected.

**3.**

We show how to concretely apply last point. We use a depth/breath-first search (DFS/BFS) to explore the graph, knowing that its complexity is $O(|V| + |E|)$.

**Algorithm 1:** Noiseless with anchor

**Input :** $\mathcal{G} = (V, E)$, measurements $(H_{ij})_{(i,j)\in E}$, and anchors $(R_j)_{j\in J}$
**Output:** Rotations $(R_i)_{i\in V}$

Choose an $a \in J$
Do a DFS or BFS of $\mathcal{G}$ starting at $a$
**for** *all search tree edges* $(i, j)$ *in order of appearance,* **do**
  $R_i = H_{ij} R_j$
**end for**
**return** $(R_i)_{i\in V}$

If there is noise, the computation done in the loop cannot be done like this since we have randomness and we don't know $Z_{ij}$. In other words, we cannot solve the problem locally edge by edge but need

to consider the whole graph and compare all measurements to estimate the rotations that gives the most likely noise.

**4.**

Recall that $\mathcal{M} = \{X \in \mathrm{SO}(d)^m | X_j = R_j \quad \forall j \in J\}$, so by defining

$$M_i = \begin{cases} \{R_i\} & \text{if } i \in J \\ \mathrm{SO}(d) & \text{else} \end{cases},$$

we can write $\mathcal{M} = \prod_{i=1}^m \mathcal{M}_i$. Then we notice that $\mathcal{M}$ is compact, considered as the product topology of compact spaces. Indeed, the singletons $\{R_i\}$ are compact and $SO(d)$ is compact because bounded and closed:

This last fact can be shown by using the fact that the maps $A \mapsto \det(A) - 1$, $A \mapsto AA^\top - I_d$, and $A \mapsto A^\top A - I_d$ are continuous in the matrices space, so pre-images of $\{0\}$ are closed, as well as their intersection $SO(d)$. The boundedness comes from the fact that $\|R\|^2 = \mathrm{Tr}(RR^\top) = \mathrm{Tr}(I_d) = d$. All functions used in the log-likelyhood are continuous, so we apply the extreme value theorem to conclude the existence of a global maximizer.

We bound the log-likelyhood by using the mode $I_d$ of $p$ and the monotonicity of the logarithm:

$$\sum_{i,j \in E} \log(p(X_i^\top H_{ij} X_j)) \leq \sum_{i,j \in E} \log(p(I_d)) = |E| \log(p(I_d))$$

with equality only if $X_i^\top H_{ij} X_j = I_d$. If there is no noise, we have equality with the true rotations. If there is no anchor neither, we can also have the solution with shifted rotations as it was showed in Question 1 so the maximizer is not unique.

**5.**

Recall that we have written $\mathcal{M} = \prod_{i=1}^m \mathcal{M}_i$ and the metric induced by the usual scalar product is indeed the product metric, so we can use results about product of manifolds.

Let's study $\mathrm{SO}(d)$. We can define it locally with the smooth map

$$\begin{cases} h: & \mathbb{R}^{d \times d} \mapsto \mathrm{Sym}(d) \\ & X \to X^\top X - I_d. \end{cases}$$

Indeed, if $X \in \mathrm{SO}(d)$, then it is invertible with inverse $X^\top$, so its determinant is positive, and it stay positive locally in the space of matrices thanks to the continuity of the determinant. As a result, for an invertible matrix $Y$ in this neighborhood, $h(Y) = 0$ imply $Y^{-1} = Y^\top$ and $XX^\top = I_p$ is also true. We know then that $\det(Y) \in \{-1, 1\}$ and since it have stayed positive, it is 1, so $Y \in \mathrm{SO}(d)$.

We compute the directional derivative in a direction matrix $U$:

$$\mathrm{D}h(X)[U] = X^\top U + U^\top X = X^\top U + (X^\top U)^\top.$$

This equal zero if and only if $X^\top U$ is a skew matrix of $\mathrm{Skew}(d)$. We can write this more simply if we parameterize $U = X\Omega$. We get $0 = X^\top X\Omega + (X^\top X\Omega)^\top = \Omega + \Omega^\top$, so the condition becomes $\Omega \in \mathrm{Skew}(d)$ and we have

$$\mathrm{T}_X \,\mathrm{SO}(d) = \ker \mathrm{D}h(X) = X \,\mathrm{Skew}(d) = \{X\Omega \in \mathbb{R}^{d \times d} | \Omega \in \mathrm{Skew}(d)\}.$$

Now we see that $\mathrm{D}h(X)$ as full rank since for any symmetric Y, $\mathrm{D}h(X)[1/2XY] = 1/2X^\top XY + 1/2(X^\top XY)^\top = 1/2Y + 1/2Y^\top = Y$. The dimension of $\mathrm{Sym}(d)$ is $d(d-1)/2$, so the dimension of the manifold $\mathrm{SO}(d)$ is $d^2 - d(d-1)/2 = d(d+1)/2$.

Finally, the singletons $\{R_j\}$ are just trivial 0-dimensional manifolds with tangent null spaces $\mathrm{T}_{R_j}\{R_j\} = \{0\}$, so $\mathcal{M} = \prod_{i=1}^m \mathcal{M}_i$ has tangent spaces

$$\mathrm{T}_X\mathcal{M} = \prod_{i=1}^m \mathrm{T}_{X_i}\mathcal{M}_i = \{U \in \mathrm{SO}(d)^m | U_i \in X_i \,\mathrm{Skew}(d) \text{ for } i \notin J \text{ and } U_i = 0 \text{ else}\},$$

and is of dimension the sum of all dimensions $\sum_{j\in J} d(d+1)/2 = |J|d(d+1)/2$.

**6.**

The orthogonal projection into a product of linear subspaces is the product of the projections, i.e.

$$\mathrm{Proj}_{\mathrm{T}_X\mathcal{M}}(U) = (\mathrm{Proj}_{\mathrm{T}_{X_i}\mathcal{M}_i}(U_i))_{i=1}^m.$$

For the trivial manifolds $\{R_j\}$ the projection to the null tangent space is the null linear map $\mathrm{Proj}_{\{0\}} \equiv 0$. For the manifolds $\mathrm{SO}(d)$ we assert that the projection $\mathrm{Proj}_{X_i\,\mathrm{Skew}(d)} : \mathbb{R}^{d\times d} \to X_i\,\mathrm{Skew}(d)$ is

$$\mathrm{Proj}_{X_i\,\mathrm{Skew}(d)}(U_i) = (U_i - XU_i^\top X)/2.$$

One can easily check that it is well defined and satisfy the orthogonality equation but let us show a constructive way to find the formula. The key is to first understand that $\mathrm{Skew}(d)$ is the complementary space of $\mathrm{Sym}(d)$:

$$\begin{aligned}\forall(\Omega, S) \in \mathrm{Skew}(d) \times \mathrm{Sym}(d),\ \langle\Omega, S\rangle &= \mathrm{Tr}(\Omega S^\top) = \mathrm{Tr}(\Omega S) = \mathrm{Tr}((\Omega S)^\top)\\ &= \mathrm{Tr}(S^\top\Omega^\top) = -\mathrm{Tr}(S\Omega) = -\mathrm{Tr}(\Omega S)\\ &= -\langle\Omega, S\rangle,\end{aligned}$$

implies orthogonality. The dimensions $d(d-1)/2$ and $d(d+1)/2$ adding up to $d^2$, we get indeed $\mathbb{R}^{d\times d} = \mathrm{Skew}(d) \oplus \mathrm{Sym}(d)$ and

$$\mathrm{Proj}_{\mathrm{Skew}(d)}(U_i) = U_i - \mathrm{Proj}_{\mathrm{Sym}(d)}(U_i) = U_i - (U_i + U_i^\top)/2 = \frac{U_i - U_i^\top}{2}.$$

Then that since the multiplication by $X_i$ is an isometry we compute

$$\begin{aligned}\mathrm{Proj}_{X_i\,\mathrm{Skew}(d)}(U_i) = \mathrm{Proj}_{X_i\,\mathrm{Skew}(d)}(X_iX_i^\top U_i) = X_i\mathrm{Proj}_{\mathrm{Skew}(d)}(X_i^\top U_i) &= X_i\frac{X_i^\top U_i - (X_i^\top U_i)^\top}{2}\\ = \frac{U_i - X_iU_i^\top X_i}{2}.&\end{aligned}$$

**7.**

We start by defining a second order retraction on $SO(d)$. To do so, observe first that $SO(d)$ is one (of the two) connected component of $O(d)$ (the orthogonal group), and therefore all the tools we develop on $O(d)$ apply just as well to $SO(d)$. Indeed, it is clear that retractions on $O(d)$ yield retractions on $SO(d)$ since, being smooth, they cannot leave a connected component. We already defined a first order retraction on $O(d)$ (see exercise section 3), which is the map $R_X : \mathcal{T}SO(d) \to SO(d)$ defined by $R_X(V) = UW^\top$, where we use the singular value decomposition $X+V = U\Sigma W^\top$. We already showed

in this exercise session that it is indeed a retraction, and that it has a useful equivalent expression $R_X(V) = (X+V)(I_d + V^\top V)^{-1/2}$.
Therefore, by the previous argument, one has the following first order retraction for $SO(d)$, which is the map $R_X : \mathcal{T}O(d) \to O(d)$ defined by $R_X(V) = UW^\top$, where we again used the singular value decomposition $X + V = U\Sigma W^\top$. We also similarly have the equivalent expression $R_X(V) = (X+V)(I_d + V^\top V)^{-1/2}$.
We still need to show that it is also a second order retraction. To do so, recall from the exercise session that $R_X(V)$ is the unique metric projection of $X + V$ to $O(d)$ (hence the restriction to $SO(d)$), that is, $Y = R_X(V)$ is the unique solution of $\min_{Y \in SO(d)} \|X + V - Y\|^2$, and therefore, by Proposition 5.55 from the text book, it induces a second order retraction on $SO(d)$, namely, $R$.
Now, the natural retractions on the singletons $\{R_j\}, j \in J$ is the map that send 0, the only vector in the tangent spaces, to the respective $R_j$, which are clearly second order retractions.
So, the retraction we suggest for $\mathcal{M}$ is the map $R : \mathcal{T}\mathcal{M} \to \mathcal{M}$ that send any $(X, V) \in \mathcal{T}\mathcal{M}$ to $R_X(V) = (R_{X_1}(V_1), R_{X_2}(V_2), \ldots, R_{X_m}(V_m))$, where $R_{X_i}(V_i) = R_i$ whenever $i \in J$, and $R_{X_i}(V_i) = (X_i + V_i)(I_d + V_i^\top V_i)^{-1/2}$ whenever $i \notin J$, which is a second order retraction as every component is a second order retraction.

**8.**

Assume that $p = p_\kappa$. For every $X \in \mathcal{M}$, we explicitly compute :

$$\begin{aligned} f(X) &= \sum_{\{i,j\} \in E} \log(p_\kappa(X_i^\top H_{ij} X_j)) \\ &= \sum_{\{i,j\} \in E} \log(\frac{1}{c_d(\kappa)} e^{\kappa \operatorname{Tr}(X_i^\top H_{ij} X_j)}) \\ &= \sum_{\{i,j\} \in E} (\log(\frac{1}{c_d(\kappa)} + \kappa \operatorname{Tr}(X_i^\top H_{ij} X_j)) \end{aligned}$$

and since $\kappa \geq 0$, finding $\hat{X}_{MLE} \in \operatorname{argmax}_{X \in \mathcal{M}} f(X)$ is equivalent to finding $\hat{X} \in \operatorname{argmax}_{X \in \mathcal{M}} (\sum_{\{i,j\} \in E} Tr(X_i^\top H_{ij} X_j))$.
We now see that :

$$\begin{aligned} \|H_{ij}X_j - X_i\|_F^2 &= \operatorname{Tr}((H_{ij}X_j - X_i)^\top (H_{ij}X_j - X_i)) \\ &= \operatorname{Tr}(X_j^\top H_{ij}^\top H_{ij} X_j - X_j^\top H_{ij}^\top X_i - X_i^\top H_{ij} X_j + X_i^\top X_i) \\ &= \operatorname{Tr}(-2X_i^\top H_{ij} X_j + 2I_d) \\ &= -2Tr(X_i^\top H_{ij} X_j) + 2d \end{aligned}$$

Where we used the facts that $X_j, X_i$ and $H_{ij} \in SO(d)$ for every $\{i,j\} \in E$.
So, in conclusion, finding $\hat{X} \in \operatorname{argmin}_{X \in \mathcal{M}} \sum_{\{i,j\} \in E} \|H_{ij}X_j - X_i\|_F^2 = \operatorname{argmin}_{X \in \mathcal{M}} \sum_{\{i,j\} \in E} (-2Tr(X_i^\top H_{ij} X_j) + 2d)$ is equivalent to finding $\hat{X} \in \operatorname{argmax}_{X \in \mathcal{M}} \sum_{\{i,j\} \in E} Tr(X_i^\top H_{ij} X_j)$, which is equivalent to finding $\hat{X}_{MLE} \in \operatorname{argmax}_{X \in \mathcal{M}} f(X)$.

# 2 Manifold factory, gradients and Hessians

**9.**

We decide to store X so that it is compatible with the manopt[1] toolbox and in particular with `rotationsfactory.m` which implement the manifold $SO(d)^m$. The variable stores only the un-

anchored part of $X$ and is then a 3D tensor of size $d \times d \times (m - m_a)$ with $m_a = |J|$. To minimize the necessary number of computations and keep the format of tangent vectors in `rotationsfactory.m`, we store only the skew part of tangent vector and will use it as inputs and outputs of functions inside the manifold. The dimension of $(mud)$ (and $SO(d)$) is $d(d+1)/2$ due to the repetition of values, but we store the skew part of tangent vectors as a normal matrix with memory $d^2$ (same for rotations) so the total memory is $(m - m_a)d^2 \in O_{m,d}(md^2)$. Notice that since the rotations will be mainly of dimension 3, the un-optimal storage of them won't be significant in contrast with $m$.

**10.**

We implement the manifold in `manyrotationsfactory.m` [1], where we represent the manifold as `rotationsfactory(d, m - m_a)`. We decide to use this manifold factory because this manifold is actually the same, up to take off the trivial dimensions which is a isometry. Like this we don't have to go inside the internal functionment of manopt, and we avoid having to debug our manifold. Instead, the changes between representation with or without anchor are all made in the problem structure itself. We take the convention to always take anchors in the first indices, so that $J = \{1, \dots, m_a\}$. The projection implemented by `manyrotationsfactory` is the same we defined, so we use it. The projector we defined is also implemented as `retr_polar` so we use it. We define a function `add_anchors` that add the anchor values to a point when needed, and a function `add_zeros` to add the zeros to tangent vectors corresponding to the anchors dimensions.

**11.**

The functions `M.rand(), M.randvec(), M.zerovec(), M.lincomb(), M.tangent()` are all already defined in `manyrotationsfactory` so we don't rewrite them.

**12.**

The scalar product define in `manyrotationsfactory` is the same as we need, so we don't rewrite it.

**13.**

Let $(X, U)$ be an element of $\mathrm{T}\mathcal{M}$. We recall that $f(X) = \sum_{\{i,j\}\in E}(\log \circ p)(X_i^\top H_{ij} X_j)$, so let us compute $\mathrm{D}f$ by using the composition rule on $\log \circ p$ and the product rule on $X_i^\top H_{ij} X_j$:

$$\mathrm{D}f(X)[U] = \sum_{\{i,j\}\in E} p(X_i^\top H_{ij} X_j)^{-1} \mathrm{D}p(X_i^\top H_{ij} X_j)[U_i^\top H_{ij} X_j + X_i^\top H_{ij} U_j]$$

Suppose $p = p_{\kappa_1,\kappa_2,q}$ , then $\mathrm{D}p = q\mathrm{D}p_{\kappa_1} + (1-q)\mathrm{D}p_{\kappa_2}$, so we compute for any $(Z, V)$ in $\mathrm{T}\,\mathrm{SO}(d)$ that

$$\mathrm{D}p_\kappa(Z)[V] = \kappa p_\kappa(Z)\,\mathrm{Tr}(V).$$

[1] the code can be found in the GitHub repository opti-manifolds-rotations-estimation[3]

giving $\mathrm{D}p_{\kappa_1,\kappa_2,q}(Z)[V] = \big(q\kappa_1 p_{\kappa_1}(Z) + (1-q)\kappa_2 p_{\kappa_2}(Z)\big)\operatorname{Tr}(V)$. We use this in the formula for $\mathrm{D}f$ and get

$$\begin{aligned}
\mathrm{D}f(X)[U] &= \sum_{\{i,j\}\in E} p(X_i^\top H_{ij} X_j)^{-1}\big(q\kappa_1 p_{\kappa_1}(X_i^\top H_{ij}X_j) + (1-q)\kappa_2 p_{\kappa_2}(X_i^\top H_{ij}X_j)\big)\operatorname{Tr}(U_i^\top H_{ij}X_j + X_i^\top H_{ij}U_j) \\
&= \sum_{\{i,j\}\in E} \frac{q\kappa_1 p_{\kappa_1}(X_i^\top H_{ij}X_j) + (1-q)\kappa_2 p_{\kappa_2}(X_i^\top H_{ij}X_j)}{p(X_i^\top H_{ij}X_j)}\operatorname{Tr}(U_i^\top H_{ij}X_j + U_j^\top H_{ij}^\top X_i) \\
&= \sum_{\{i,j\}\in E} P_{ij}(\langle H_{ij}X_j, U_i\rangle + \langle H_{ji}X_i, U_j\rangle) \quad \text{with } P_{ij} = \frac{q\kappa_1 p_{\kappa_1}(X_i^\top H_{ij}X_j) + (1-q)\kappa_2 p_{\kappa_2}(X_i^\top H_{ij}X_j)}{p(X_i^\top H_{ij}X_j)} \\
&= \sum_{i\in V}\sum_{j:\{i,j\}\in E} P_{ij}\langle H_{ij}X_j, U_i\rangle \\
&= \sum_{i\in V}\left\langle \sum_{j:\{i,j\}\in E} P_{ij}H_{ij}X_j, U_i\right\rangle \\
&= \left\langle \left(\sum_{j:\{i,j\}\in E} P_{ij}H_{ij}X_j\right)_{i\in V}, U\right\rangle_X .
\end{aligned}$$

By identification,

$$\nabla f(X) = \left(\sum_{j:\{i,j\}\in E} P_{ij}H_{ij}X_j\right)_{i\in V},$$

and we can project it to the tangent space to obtain the Riemannian gradient. We do it component by component using Question 6:

$$\operatorname{grad} f(X)_i = \operatorname{Proj}_{X_i}\nabla f(X)_i = X_i \operatorname{Skew}(X_i^\top \nabla f(X)_i) = X_i \operatorname{Skew}\left(\sum_{j:\{i,j\}\in E} P_{ij}X_i^\top H_{ij}X_j\right)$$

if $i \notin J$. Else, $\operatorname{grad} f(X)_i = 0$.

When $p = p_\kappa = p_{\kappa,0,0}$, we get that $P_{ij} = \kappa$ and

$$\operatorname{grad} f(X)_i = \kappa X_i \operatorname{Skew}\left(\sum_{j:\{i,j\}\in E} X_i^\top H_{ij}X_j\right)$$

**14.**

Let $(X,U)\in\mathcal{TM}$. For every $i,j\in V$ we define $Z_{ij} = X_i^\top H_{ij}X_j$. Note that we redefine some notation from the previous question i.e $P_{ij} := P(Z_{ij}) = \frac{q\kappa_1 p_{\kappa_1}(Z_{ij}) + (1-q)\kappa_2 p_{\kappa_2}(Z_{ij})}{p(Z_{ij})}$.
Now, for unanchored node, i.e $i \notin J$, we define the map $\overline{\operatorname{grad}_i f} : (\mathbb{R}^{d\times d})^m \to \mathbb{R}^{d\times d}$ by $\overline{\operatorname{grad}_i f}(X) = X_i \sum_{j:\{i,j\}\in E} P_{ij} \operatorname{Skew}(Z_{ij})$. We know that the restriction of $\overline{\operatorname{grad}_i f}$ to $\mathcal{M}$ yields the $i$-th component of the gradient of $f$ by the previous question. Therefore, the $i$-th component of the Hessian of $f$ at $X$ applied to the tangent vector $U = X\Omega$, where $\Omega = (\Omega_1,\dots,\Omega_m)$ with $\Omega_i \in \operatorname{Skew}(d)$ for every $i \notin J$ and $\Omega_i = 0$ otherwise, is given by $\operatorname{Hess}_i f(X)[X\Omega] = X_i \operatorname{Skew}(X_i^\top D\overline{\operatorname{grad}_i f}(X)[X\Omega])$, since $\operatorname{Hess} f(X)[U] = \nabla_u \operatorname{grad} f = Proj_X(D\overline{\operatorname{grad} f}(X)[U])$, where $\nabla_u$ is the unique Riemannian connection on $\mathcal{M}$, and $\overline{\operatorname{grad} f}$ is an extension of the gradient of $f$ to the linear space $(\mathbb{R}^{d\times d})^m$.
We now want to explicitly compute $X_i^\top D\overline{\operatorname{grad}_i f}(X)[X\Omega]$ for unanchored nodes $i$. Using the product

rule and the linearity of the derivation we get :

$$\begin{aligned}
X_i^\top D\overline{\text{grad}_i f}(X)[X\Omega] &= X_i^\top \sum_{j:\{i,j\}\in E} D(X \mapsto X_i P(Z_{ij}) \operatorname{Skew}(Z_{ij}))[X\Omega] \\
&= X_i^\top \sum_{j:\{i,j\}\in E} D(X \mapsto X_i)[X\Omega] P(Z_{ij}) \operatorname{Skew}(Z_{ij}) + X_i D(X \mapsto P(Z_{ij}) \operatorname{Skew}(Z_{ij}))[X\Omega] \\
&= \sum_{j:\{i,j\}\in E} X_i^\top D(X \mapsto X_i)[X\Omega] P(Z_{ij}) \operatorname{Skew}(Z_{ij}) + D(X \mapsto P(Z_{ij}))[X\Omega] \operatorname{Skew}(Z_{ij}) \\
&+ P(Z_{ij}) D(X \mapsto \operatorname{Skew}(Z_{ij}))[X\Omega]
\end{aligned}$$

We now compute the 3 different derivatives :

$$\begin{aligned}
D(X \mapsto X_i)[X\Omega] &= \lim_{t\to 0} \frac{1}{t}(X_i + tX_i\Omega_i - X_i) \\
&= X_i\Omega_i
\end{aligned}$$

and then, using the composition rule :

$$D(X \mapsto \operatorname{Skew}(Z_{ij}))[X\Omega] = D \operatorname{Skew}(Z_{ij})[D(X \mapsto Z_{ij})[X\Omega]]$$

where :

$$\begin{aligned}
D(X \mapsto Z_{ij})[X\Omega] &= \lim_{t\to 0} \frac{1}{t}((X_i + tX_i\Omega_i)^\top H_{ij}(X_j + tX_j\Omega_j) - X_i^\top H_{ij} X_j) \\
&= \lim_{t\to 0} \frac{1}{t}(t\Omega_i^\top X_i^\top H_{ij} X_j + tX_i^\top H_{ij} X_j\Omega_j + t^2\Omega_i^\top X_i^\top H_{ij} X_j\Omega_j) \\
&= \Omega_i^\top X_i^\top H_{ij} X_j + X_i^\top H_{ij} X_j\Omega_j \\
&= Z_{ij}\Omega_j - \Omega_i Z_{ij} =: \Omega_{ij}
\end{aligned}$$

So now, we get :

$$\begin{aligned}
D(X \mapsto \operatorname{Skew}(Z_{ij}))[X\Omega] &= D \operatorname{Skew}(Z_{ij})[\Omega_{ij}] \\
&= \lim_{t\to 0} \frac{1}{t}(\operatorname{Skew}(Z_{ij} + t\Omega_{ij}) - \operatorname{Skew}(Z_{ij})) \\
&= \operatorname{Skew}(\Omega_{ij})
\end{aligned}$$

Finally, we compute :

$$D(X \mapsto P(Z_{ij}))[X\Omega] = DP(Z_{ij})[\Omega_{ij}]$$

and now, using again the product rule and the definition of $P(Z_{ij})$, we have:

$$\begin{aligned}
DP(Z_{ij})[\Omega_{ij}] &= D(Z_{ij} \mapsto q\kappa_1 p_{\kappa_1}(Z_{ij}) + (1-q)\kappa_2 p_2(Z_{ij}))[\Omega_{ij}] \frac{1}{p(Z_{ij})} \\
&+ (q\kappa_1 p_{\kappa_1}(Z_{ij}) + (1-q)\kappa_2 p_2(Z_{ij})) D(Z_{ij} \mapsto \frac{1}{p(Z_{ij})})[\Omega_{ij}]
\end{aligned}$$

Now we get :

$$\begin{aligned}
D(Z_{ij} \mapsto q\kappa_1 p_{\kappa_1}(Z_{ij}) + (1-q)\kappa_2 p_2(Z_{ij}))[\Omega_{ij}] &= q\kappa_1 D p_{\kappa_1}(Z_{ij})[\Omega_{ij}] + (1-q)\kappa_2 D p_{\kappa_2}(Z_{ij})[\Omega_{ij}] \\
&= q\kappa_1^2 p_{\kappa_1}(Z_{ij}) Tr(\Omega_{ij}) + (1-q)\kappa_2^2 p_{\kappa_2}(Z_{ij}) Tr(\Omega_{ij}) \\
&= (q\kappa_1^2 p_{\kappa_1}(Z_{ij}) + (1-q)\kappa_2^2 p_{\kappa_2}(Z_{ij})) Tr(\Omega_{ij})
\end{aligned}$$

and for the second derivative, we define $F : \mathbb{R}^* \to \mathbb{R}^*$ by $F(x) = \frac{1}{x}$, so that we can use the chain rule:

$$\begin{aligned} D(Z_{ij} \mapsto \frac{1}{p(Z_{ij})})[\Omega_{ij}] &= DF \circ p(Z_{ij})[\Omega_{ij}] \\ &= DF(p(Z_{ij})[Dp(Z_{ij})[\Omega_{ij}]] \\ &= DF(p(Z_{ij})[(q\kappa_1 p_{\kappa_1}(Z_{ij}) + (1-q)\kappa_2 p_{\kappa_2}(Z_{ij}))Tr(\Omega_{ij})] \\ &= -\frac{q\kappa_1 p_{\kappa_1}(Z_{ij}) + (1-q)\kappa_2 p_{\kappa_2}(Z_{ij})}{p(Z_{ij})^2} Tr(\Omega_{ij}) \end{aligned}$$

and combining these two results together yield :

$$DP(Z_{ij})[\Omega_{ij}] = (\frac{q\kappa_1^2 p_{\kappa_1}(Z_{ij}) + (1-q)\kappa_2^2 p_{\kappa_2}(Z_{ij})}{p(Z_{ij})} - P(Z_{ij})^2) Tr(\Omega_{ij})$$

and finally, we get :

$$\begin{aligned} X_i^\top D\overline{\mathrm{grad}_i f}(X)[X\Omega] = &\sum_{j:\{i,j\}\in E} \Omega_i P(Z_{ij}) \operatorname{Skew}(Z_{ij}) \\ &+ (\frac{q\kappa_1^2 p_{\kappa_1}(Z_{ij}) + (1-q)\kappa_2^2 p_{\kappa_2}(Z_{ij})}{p(Z_{ij})} - P(Z_{ij})^2) Tr(\Omega_{ij}) \operatorname{Skew}(Z_{ij}) + P(Z_{ij}) \operatorname{Skew}(Z_{ij}) \\ \implies \mathrm{Hess}_i f(X)[X\Omega] = X_i Skew\Bigg( &\sum_{j:\{i,j\}\in E} \Big( \Omega_i P(Z_{ij}) \operatorname{Skew}(Z_{ij}) \\ &+ (\frac{q\kappa_1^2 p_{\kappa_1}(Z_{ij}) + (1-q)\kappa_2^2 p_{\kappa_2}(Z_{ij})}{p(Z_{ij})} - P(Z_{ij})^2) Tr(\Omega_{ij}) \operatorname{Skew}(Z_{ij}) \\ &+ P(Z_{ij}) \operatorname{Skew}(\Omega_{ij}) \Big) \Bigg) \end{aligned}$$

Finally, observe that for $i \in J$, the $i$-th component of the Hessian simply vanishes.

**15.**

Recall that the formula of the gradient is

$$\mathrm{grad} f(X)_i = \mathrm{Proj}_{X_i} \nabla f(X)_i = X_i \operatorname{Skew}(X_i^\top \nabla f(X)_i) = X_i \operatorname{Skew}\left( \sum_{j:\{i,j\}\in E} P_{ij} X_i^\top H_{ij} X_j \right)$$

for indices $i$ outside of $J$. To compute the products $Z_{ij} = X_i^\top H_{ij} X_j$ we need $|E|2d^3$ (plus $|E|2d^3$ additions). This is done by

```
Z = pagemtimes(X(:,:,data.I),'transpose', data.H,'none'); Z = pagemtimes(Z, X(:,:,data.J));
```

The we compute the fraction $P$ by first computing $etz_{ij} = e^{\mathrm{Tr}\, Z_{ij}}$(so that we minimize the computations) which takes $|E|$ (plus $d|E|$ additions). The expression of the fraction is then `(q*k1/c1 * ... etz.^k1 + (1-q)*k2/c2 * etz.^k2)/(q/c1 * etz.^k1 + (1-q)/c2 * etz.^k2)` and takes $14|E|$ operations ($4|E|$ additions). The multiplication between $P$ and $Z$ is done in $d^2|E|$ (and the sums takes less than $2d^2|E|$ additions). At then end, taking the skew part involve $d^2|E|$ arithmetic operations ($d^2|E|$ additions).

We sum up everything and get

$$|E|(2d^3 + 2d^2 + 15)$$

operations (and $|E|(2d^3 + 3d^2 + d + 4)$ additions), so

$$|E|(4d^3 + 5d^2 + d + 19)$$

if we count additions. For $d = 3$, this is $175|E|$.

The formula of the Hessian is just over there in Question 14. Again, computing $Z$ takes $|E|2d^3$ (plus $|E|2d^3$ additions) and computing $etz$ take $|E|$ (plus $d|E|$ additions). Using this variable, $P$ takes $14|E|$ operations ($4|E|$ additions) and the fraction in the second term takes $16|E|$ operations ($4|E|$ additions). The matrix $\Omega_{ij} = Z_{ij}\Omega_j - \Omega_i Z_{ij}$ takes $|E|2d^3$ ($|E|4d^3 + d^2$), taking the trace takes $|E|d$ additions. Computing $skew(Z_ij)$ takes $d^2|E|$ arithmetic operations ($d^2|E|$ additions). Lets compute the compelexity of each internal terms. The first takes $|E|(d^2 + d^3)(|E|d^3$ additions) the second $|E|(2 + d^2)$ ($|E|$ additions), and the last takes $|E|d^2$ operations. Computing the big sums takes less than $d^2|E|$ additions, and the mu$takes$d$^2|E|$ arithmetic operations ($d^2|E|$ additions).

We sum everything and get $|E|(5d^3 + 5d^2 + 33)$ (and $|E|(7d^3 + 3d^2 + 2d + 1)$) so

$$|E|(12d^3 + 8d^2 + 2d + 34)$$

in total with additions. For $d = 3$, this is $436|E|$, so approximately 2.5 times more than gradient. The trick is that we don't actually compute the Hesssian, but only its evaluation as a linear map for a certain entry. To compute the value of the Hessian like if it was a classic matrix Hessian, it would need to compute $\text{Hess}_i f(X)[X\Omega^{(b)}]$ for the $||d(d-1)/2$

**16.**

We implement the cost in `cost.m` and the riemanian gradient in `grad.m`, and the riemanian hessian in `hessian.m`. We also impement a function that compute the cost and the gradient together to avoid computing things multiple times. This is done in `costgrad.m`. We build some data and problem structre and use the manopt debugging functions `checkgradient` and `checkhessian` and obtain the plot in Figure [1]. This run and all running parts are done in the file `main.m` with cells runnable separately.

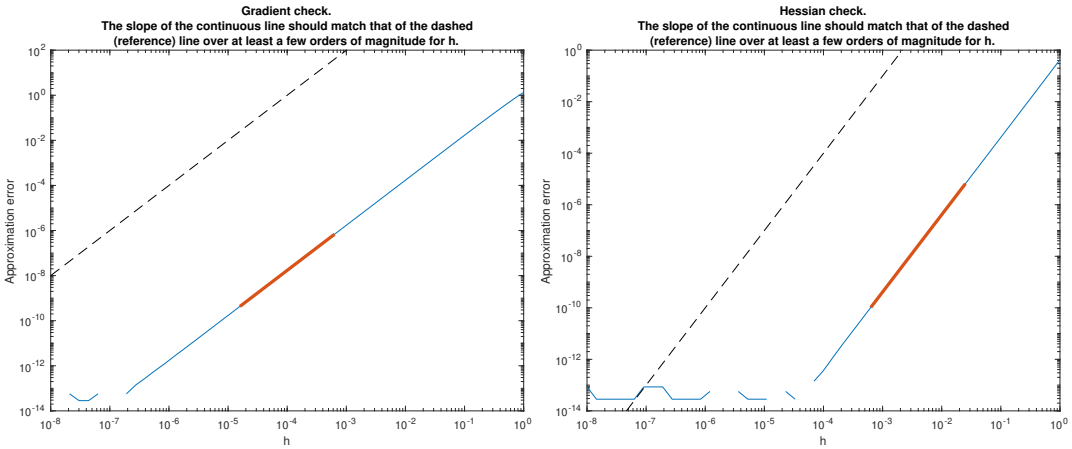


Figure 1: Sanity check: gradient and hessian

We see that we indeed get the first and second order approximation. The printed output is

```
          Question 16

Check Gradient:
The slope should be 2. It appears to be: 1.99985.
If it is far from 2, then directional derivatives might be erroneous.
The residual should be 0, or very close. Residual: 0.
If it is far from 0, then the gradient is not in the tangent space.
In certain cases (e.g., hyperbolicfactory), the tangency test is inconclusive.

Check Hessian:
The slope should be 3. It appears to be: 3.00018.
If it is far from 3, then directional derivatives,
the gradient or the Hessian might be erroneous.
Tangency residual should be zero, or very close; residual: 0.
If it is far from 0, then the Hessian is not in the tangent space.
||a*H[d1] + b*H[d2] - H[a*d1+b*d2]|| should be zero, or very close.
    Value: 2.98033e-15 (norm of H[a*d1+b*d2]: 11.855)
If it is far from 0, then the Hessian is not linear.
<d1, H[d2]> - <H[d1], d2> should be zero, or very close.
    Value: -0.0188979 - -0.0188979 = -1.11022e-15.
If it is far from 0, then the Hessian is not symmetric.
```

and confirm they both work fine.

# 3 Generating data

# 4 Algorithms

## 4.1 Initialization

**17.**

For any $Y \in \mathbb{R}^{dm\times d}$ such that $Y = (Y_1, \dots, Y_m)^\top$ where $Y_j \in SO(d), \forall j = 1, \dots, m$, we have that $M := W_1 Y$ is a block matrix in $\mathbb{R}^{dm\times d}$ where each block $M_i \in \mathbb{R}^{d\times d}$ is defined by :

$$M_i = \sum_{j=1}^{m} W_{i,j} Y_j$$

Therefore, $Y^\top W_1 Y$ is a $d \times d$ matrix defined by :

$$\begin{aligned} Y^\top W_1 Y = Y^\top M &= \sum_{i=1}^{m}\sum_{j=1}^{m} Y_i^\top W_{i,j} Y_j \\ &= \sum_{\{i,j\}\in E} Y_i^\top H_{ij} Y_j \end{aligned}$$

Thus, finding $\hat{Y} \in \operatorname{argmax}_{Y\in\mathbb{R}^{dm\times d}} \operatorname{Tr}(Y^\top W_1 Y)$ subject to $Y_j \in SO(d), \forall j = 1, \dots, m$ is equivalent to finding $\hat{Y} \in \operatorname{argmax}_{Y\in\mathbb{R}^{dm\times d}} \sum_{\{i,j\}\in E} Tr(Y_i^\top H_{ij} Y_j)$ subject to $Y_j \in SO(d), \forall j = 1, \dots, m$.
Now, recall from the solution of question 8 that whenever $p = p_\kappa$, computing the maximum likelihood estimator is equivalent to finding $\hat{X}_{MLE} \in \operatorname{argmax}_{X\in\mathcal{M}} \sum_{\{i,j\}\in E} Tr(X_i^\top H_{ij} X_j)$. But, whenever there are no anchor, we have that $\mathcal{M} = (SO(d))^m$, and we conclude by observing that in this case, finding $\hat{Y} \in \operatorname{argmax}_{Y\in\mathbb{R}^{dm\times d}} \sum_{\{i,j\}\in E} Tr(Y_i^\top H_{ij} Y_j)$ subject to $Y_j \in SO(d), \forall j = 1, \dots, m$ is equivalent to finding $\hat{X}_{MLE} \in \operatorname{argmax}_{X\in\mathcal{M}} \sum_{\{i,j\}\in E} Tr(X_i^\top H_{ij} X_j)$.

**18.**

Assume that $Y_j \in SO(d)$ for all $j = 1, \dots, m$.
Observe first that by the definition of the Kronecker product, $D_1 = D \otimes I_d$ is a $dm \times dm$ matrix that can be seen as an $m \times m$ block matrix where each block is a $d \times d$ matrix, and moreover, all the blocks are zero matrices, except on the main diagonal, and they are of the form $deg(i)I_d$.
Using this observation we now compute :

$$\begin{aligned} & D_1 Y = (deg(1)Y_1, \dots, deg(m)Y_m)^\top \\ \implies & Y^t D_1 Y = \sum_{i=1}^{m} deg(i) Y_i^\top Y_i \\ & = \sum_{i=1}^{m} deg(i) I_d \\ & = \mathrm{Tr}(D) I_d \end{aligned}$$

**19.**

a) Observe that $Y^\top D_1 Y = \mathrm{Tr}(D) I_d \iff \frac{1}{Tr(D)} Y^\top D_1 Y = I_d$. Therefore, by defining $D_1^{1/2}$ whose entries are the square roots of the entries of the (diagonal) matrix $D_1$, which by definition are all non-negative, we suggest the change of variables $Z = \frac{1}{\sqrt{Tr(D)}} D_1^{1/2} Y \in \mathbb{R}^{dm \times d}$.
Now, we have that $Y^\top D_1 Y = \mathrm{Tr}(D) I_d \iff Z^\top Z = I_d \iff Z \in St(dm, d)$. Moreover, one can see that $Y = \sqrt{Tr(D)} D_1^{-1/2} Z$, where $D_1^{-1/2}$ is obtained by inverting the entries of $D_1^{1/2}$.
Therefore, we can observe that :

$$\begin{aligned} \max_{Y \in \mathbb{R}^{dm \times d} : Y^\top D_1 Y = Tr(D) I_d} \mathrm{Tr}(Y^\top W_1 Y) &= \max_{Z \in St(dm,d)} Tr(Tr(D) Z^\top D_1^{-1/2} W_1 D_1^{-1/2} Z) \\ &= \max_{Z \in St(dm,d)} Tr(Z^\top D_1^{-1/2} W_1 D_1^{-1/2} Z) \end{aligned}$$

One can already see that this form of our problem looks familiar. Indeed, one can relate this formulation to the PCA formulation.
Now, if we let $Y \in \mathbb{R}^{dm \times d}$ be such that $Y^\top D_1 Y = Tr(D) I_d$ and the $d$ columns of $Y$ form a dominant set of generalized eigenvectors of the GEP with pencil $(W_1, D_1)$, one can see that if we introduce $\Lambda \in \mathbb{R}^{d \times d}$ be a diagonal matrix whose entries are the corresponding generalized eigenvalues, then we have that :

$$\begin{aligned} & W_1 Y = D_1 Y \Lambda \\ \implies & \sqrt{Tr(D)} W_1 D_1^{-1/2} Z = \sqrt{Tr(D)} D_1^{-1/2} Z \Lambda \\ \implies & D_1^{-1/2} W_1 D_1^{-1/2} Z = Z \Lambda \end{aligned}$$

So in this case, the columns of $Z$ are eigenvectors of the eigenvalue problem $D_1^{-1/2} W_1 D_1^{-1/2} v = \lambda v$ associated to the corresponding eigenvalues given by the diagonal entries of $\Lambda$. That is, $Z$ is formed by collecting the $d$ top eigenvectors (recall from the definition of dominant set of generalized eigenvectors that these consist of eigenvectors whose corresponding eigenvalues are at least as large as all other generalized eigenvalues of the GEP) of $D_1^{-1/2} W_1 D_1^{-1/2}$, and we know that this yields a global optimum for $\max_{Z \in St(dm,d)} Tr(Z^\top D_1^{-1/2} W_1 D_1^{-1/2} Z)$.
So, in other words, $Y$ is a global maximizer of (7).

b) We can use Matlab's "eigs" function to return the top $d$ eigenvalues of $D_1^{-1/2} W_1 D_1^{-1/2}$ in the forme of a diagonal matrix whose entries are those eigenvalues, and a matrix $Z$ whose columns are the corresponding eigenvectors. We can then find back our global maximizer $Y$ using the relation between $Z$ and $Y$ detailed previously. Note that in the case where the eigenvectors returned by Matlab's function are not normalized, we will need to normalize them in order to form a matrix $Z$ that is in the Stiefel Manifold. Observe also that Matlab can technically solve the GEP (using the command `[Y, ] = eigs(W1, D1, d))` we would like to solve, and this could be another way of computing our global maximizer.

**20.**

a) Observe that by definition, the $d$ columns of $Y \in \mathbb{R}^{dm \times d}$ (satisfying $Y^\top D_1 Y = \mathrm{Tr}(D) I_d$) form a dominant set of generalized eigenvectors of the GEP with pencil $(W_1, D_1)$ if and only if $W_1 Y = D_1 Y \Lambda$ where $\Lambda \in \mathbb{R}^{d \times d}$ is a diagonal matrix whose entries are the corresponding generalized eigenvalues. Note that if we assume that we have a complete measurement graph and there is no noise, we have that $D_1 = (m-1) I_{dm}$, and $W_1 = \mathcal{R}\mathcal{R}^\top$. Therefore, we have that :

$$\begin{aligned} & W_1 Y = D_1 Y \Lambda \\ \iff & \mathcal{R}\mathcal{R}^\top Y = (m-1) Y \Lambda \\ & \frac{1}{m-1} \mathcal{R}\mathcal{R}^\top Y = Y \Lambda \end{aligned}$$

where in the equation $\frac{1}{m-1}\mathcal{R}\mathcal{R}^\top Y = Y\Lambda$ we recognize an eigenvalue problem, that is, $\frac{1}{m-1}\mathcal{R}\mathcal{R}^\top Y = Y\Lambda \iff$ the columns of $Y$ are eigenvectors of $\frac{1}{m-1}\mathcal{R}\mathcal{R}^\top$ and the diagonal entries of $\Lambda$ are the associated top $d$ eigenvalues.
Note that the eigenvalues of $\frac{1}{m-1}\mathcal{R}\mathcal{R}^\top$ are the eigenvalues of $\frac{1}{m-1}\mathcal{R}^\top\mathcal{R}$, plus additional zeros (since $\mathcal{R}\mathcal{R}^\top$ is of a bigger size than $\mathcal{R}^\top\mathcal{R}$). Now, we see that $\frac{1}{m-1}\mathcal{R}^\top\mathcal{R} = \frac{m}{m-1} I_d$, so the top $d$ eigenvalues of $\frac{1}{m-1}\mathcal{R}\mathcal{R}^\top$ are all equal to $\frac{m}{m-1}$.
It now follows that :

$$\begin{aligned} & \frac{1}{m-1}\mathcal{R}\mathcal{R}^\top Y = Y\Lambda \\ \iff & \mathcal{R}\mathcal{R}^\top Y = mY \end{aligned}$$

Therefore, showing that the $d$ columns of $Y \in \mathbb{R}^{dm \times d}$ (satisfying $Y^\top D_1 Y = \mathrm{Tr}(D) I_d$) form a dominant set of generalized eigenvectors of the GEP with pencil $(W_1, D_1)$ if and only if $Y = \mathcal{R}Q$ for some orthogonal matrix $Q \in \mathbb{R}^{d \times d}$, is equivalent to showing that for $Y \in \mathbb{R}^{dm \times d}$ satisfying $Y^\top D_1 Y = \mathrm{Tr}(D) I_d$, we have that $\mathcal{R}\mathcal{R}^\top Y = mY \iff Y = \mathcal{R}Q$ for some orthogonal matrix $Q \in \mathbb{R}^{d \times d}$.
So assume first that $\mathcal{R}\mathcal{R}^\top Y = mY$ holds. Then, if we define $Q = \frac{1}{m}\mathcal{R}^\top Y \in \mathbb{R}^{d \times d}$, we have that :

$$\begin{aligned} Q^\top Q &= \frac{1}{m^2} Y^\top \mathcal{R}\mathcal{R}^\top Y \\ &= \frac{1}{m} Y^\top Y \end{aligned}$$

but since we assumed $Y^\top D_1 Y = \mathrm{Tr}(D) I_d$, i.e $(m-1) Y^\top Y = m(m-1) I_d$, it follows that $Q^\top Q = I_d$ and hence, is orthognal as wanted.

Assume now that $Y = \mathcal{R}Q$ for some orthogonal matrix $Q \in \mathbb{R}^{d\times d}$. Then we compute :

$$\begin{aligned}\mathcal{R}\mathcal{R}^\top Y &= \mathcal{R}\mathcal{R}^\top \mathcal{R}Q \\ &= m\mathcal{R}Q \\ &= mY\end{aligned}$$

and we are done.

b) Let $Y$ be a valid solution to (7), that is, $Y \in \text{argmax}_{Y\in\mathbb{R}^{dm\times d}} \text{Tr}(YW_1Y)$ subject to $Y^\top D_1 Y = \text{Tr}(D)I_d$. Part (a) combined with question 19 shows that each block $Y_j$ belong to $O(d)$ in these settings. But we don't know whether there determinant equals 1 or $-1$.
Observe that by part (a), we can decompose each block as $Y_j = R_jQ$, where $Q \in O(d)$. It follows that $\forall j = 1,\dots,m$, $\det(Y_j) = \det(R_j)\det(Q) = \det(Q)$. In other words, either all the blocks $Y_j$ belong to $SO(d)$ (have determinant equal to 1), or none of them belong to $SO(d)$ (they all have determinant equal to $-1$).
In the first case, $Y^{(a)} = Y$ is a valid solution to (6) since it belongs to $SO(d)$ by assumption and maximizes the same cost function. Note that the objective value to (6) is necessarily smaller or equal to the objective value of (7). Indeed, it follows from the fact that we maximize the same cost function, but on different sets, but we have that $\{Y \in \mathbb{R}^{dm\times d} | Y_j \in SO(d), \forall j = 1,\dots,m\} \subseteq \{Y \in \mathbb{R}^{dm\times d} | Y^\top D_1 Y = \text{Tr}(D)I_d\}$.
Now, in the other case, we see that $\forall j = 1,\dots,m$, $\det(Y_j^{(b)}) = \det(Y_j)\det(\mathcal{J}) = (-1)^2 = 1$ and that ${Y_j^{(b)}}^\top Y_j^{(b)} = \mathcal{J}^\top Y_j^\top Y_j \mathcal{J} = \mathcal{J}^\top\mathcal{J} = I_d$, hence, $Y_j^{(b)} \in SO(d)$.
So, we finally argue that $Tr(Y^\top W_1 Y) = \text{Tr}({Y^{(b)}}^\top W_1 Y^{(b)})$ in order to conclude this question. Indeed, on one side we see that :

$$\begin{aligned}\text{Tr}(Y^\top W_1 Y) &= \text{Tr}(Q^\top \mathcal{R}^\top \mathcal{R}\mathcal{R}^\top \mathcal{R}Q) \\ &= \text{Tr}(m^2 I_d) \\ &= m^2 d\end{aligned}$$

and on the other side :

$$\begin{aligned}\text{Tr}({Y^{(b)}}^\top W_1 Y^{(b)}) &= \text{Tr}(\mathcal{J}^\top Y^\top W_1 Y \mathcal{J}) \\ &= \text{Tr}(m^2 \mathcal{J}^\top \mathcal{J}) \\ &= \text{Tr}(m^2 I_d) \\ &= m^2 d\end{aligned}$$

**21.**

We have by definition that $\prod_{SO(d)}(\sum_{j\in J} \tilde{X}_j^\top R_j) \in \text{argmin}_{Q\in SO(d)} \|Q - \sum_{j\in J} \tilde{X}_j^\top R_j\|_F^2$. Now, we see that :

$$\begin{aligned}\|Q - \sum_{j\in J} \tilde{X}_j^\top R_j\|_F^2 &= \text{Tr}((Q - \sum_{j\in J} \tilde{X}_j^\top R_j)^\top (Q - \sum_{j\in J} \tilde{X}_j^\top R_j)) \\ &= \text{Tr}(Q^\top Q - \sum_{j\in J} Q^\top \tilde{X}_j^\top R_j - \sum_{j\in J} R_j^\top \tilde{X}_j Q + (\sum_{j\in J} R_j^\top \tilde{X}_j)(\sum_{j\in J} \tilde{X}_j^\top R_j)) \\ &= d - \sum_{j\in J} Tr(Q^\top \tilde{X}_j^\top R_j + R_j^\top \tilde{X}_j Q) + Tr((\sum_{j\in J} R_j^\top \tilde{X}_j)(\sum_{j\in J} \tilde{X}_j^\top R_j))\end{aligned}$$

So, it follows that $\prod_{SO(d)}(\sum_{j\in J}\tilde{X}_j^\top R_j) \in \text{argmax}_{Q\in SO(d)}\sum_{j\in J} Tr(Q^\top \tilde{X}_j^\top R_j + R_j^\top \tilde{X}_j Q)$.
On the other side, we see that :

$$\begin{aligned}\sum_{j\in J}\|R_j - \tilde{X}_j Q\|_F^2 &= \sum_{j\in J} Tr((R_j - \tilde{X}_j Q)^\top (R_j - \tilde{X}_j Q)) \\ &= \sum_{j\in J} Tr(R_j^\top R_j - R_j^\top \tilde{X}_j Q - Q^\top \tilde{X}_j^\top R_j + Q^\top \tilde{X}_j^\top \tilde{X}_j Q) \\ &= \sum_{j\in J}(Tr(I_d) - \text{Tr}(R_j^\top \tilde{X}_j Q + Q^\top \tilde{X}_j^\top R_j) + Tr(Q^\top Q)) \\ &= \sum_{j\in J}(2d - \text{Tr}(R_j^\top \tilde{X}_j Q + Q^\top \tilde{X}_j^\top R_j))\end{aligned}$$

And we observe that finding $\hat{X} \in \text{argmin}_{Q\in SO(d)}\sum_{j\in J}\|R_j - \tilde{X}_j Q\|_F^2$ is equivalent to finding $\hat{X} \in \text{argmax}_{Q\in SO(d)}\sum_{j\in J}\text{Tr}(R_j^\top \tilde{X}_j Q + Q^\top \tilde{X}_j^\top R_j)$, so, in other words, $\prod_{SO(d)}(\sum_{j\in J}\tilde{X}_j^\top R_j) \in \text{argmin}_{Q\in SO(d)}\sum_{j\in J}\|R_j - \tilde{X}_j Q\|_F^2$.

**22.**

We write the code of this initialization in `initialization.m`. To test it, we write assertions along the code. We assert the solution `Y` of the GEP is right with

```
assert(norm(W*Y-D1*Y*S)< tol),
```

we assert that $Y$ is also feasible for the constraints of the minimization problem with

```
assert(norm(Y'*D1*Y - trace(D)*eye(d))< tol).
```

We also assert that the projections of matrices `A` to $SO(d)$ gives orthogonal rotation matrices `R` with

```
assert(norm(R'*R - eye(size(A)))< tol); assert(abs(det(R)- 1)< tol);.
```

To check if it gives a candidate with a better cost (cost is the negative of the loglikelyhood) that a random one, we sample some points and compare the average and standard deviation of the cost. This is done in `main.m` at cell `Question 22` and we get the following input in the Command Window:

```
1            Question 22
2  Initial cost : -681.418514
3  Random cost  : -215.384533 +- 34.516908
```

This confirms that the spectral initialization is indeed a good first guess.

## 4.2 Riemannian gradient descent and trust-regions

**23.**

Like proposed in [2, Section 6.4.6], we take an estimation of the diameter of the manifold for $\bar{\Delta}$. To get this estimation, we take the double of the norm of the vector space, which is $2\|X\| = 2\sqrt{(m-ma)\|X_{11}\|^2} = 2\sqrt{(m-ma)d}$. According to this value, we choose $\Delta_0 = \bar{\Delta}/8$.

Like in Question 22, we test the MSE on random data and on the initialization and we get

```
           Question 23
Initial MSE          : 10.742230
Random MSE           : 10.799410 +- 1.843152
Expected random MSE : 10.5797 (= 2/3 * pi^2 + 4)
```

So we see that the spectral guess is a good candidate to optimize the objectif function, the only "error" that we have, but is not a better candidate for the actual real error.

**24.**

a) We initialize the problem in `main.m`, at cell `Question 22` with `prob = build_problem(d, ... m, ma, kappa1, kappa2, q, G);`.

b) We choose the random point with `X0 = prob.M.rand();`.

c) We run RGD and RTR using the options that we defined in `prob.option` during `build_problem`:

```
problem.option.Delta_bar = 2*sqrt(d*(m-ma));
problem.option.Delta0 = problem.option.Delta_bar / 8;
problem.option.statsfun = statsfunhelper('mse', @(X) problem.MSE(X));
problem.option.tolgradnorm = 1e-6 / problem.cardE;
```

The gradient norm tolerance is $10^{-6}$ normalized by $|E|$ since $f$ is a sum of $|E|$ terms.

d) We plot the gradient norm and the objective function in Figure [2] with a log scale in the abscissa. We see that RGD gradient norm has a linear convergence of rate $e^{-0.489753} = 0.6128$, when TRG

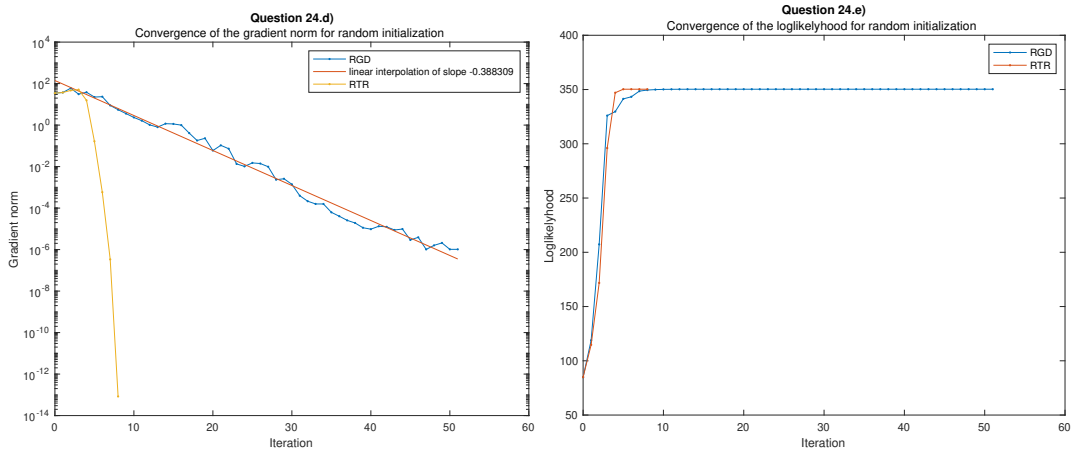


Figure 2: Question 24 d),e)

takes way less iterations to converge, and looks super-linear (confirming the local super-linearity convergence of RTR stated in [2, Theorem 6.30]), and we have 40 iterations versus 10. However, the actual running time per iteration is highly increased for TRG due to the inner optimization problem. The outputs of the methods printed are

```
           Question 24
      Random initialization
```

```
RGD:
Last stepsize smaller than minimum allowed; options.minstepsize = 1e-10.
Total time is 0.105489 [s] (excludes statsfun)
RTR:
Gradient norm tolerance reached; options.tolgradnorm = 1e-07.
Total time is 0.417930 [s] (excludes statsfun)
```

So we see that RGD is actually faster, but that it failed to reach the tolerance because steps were too small. TRG becomes interesting at the end when RGD steps size become too small.

e) Concerning the objective function, we see that the value quickly reach a step value, and we dont see it change at this scale, so we decide to substract this step, take the negative, and plot with a log scale in the abcissa in Figure [3]. We also plot the cost value of the true rotations, and we see that it is over the minimum computed. This is because of the model, it contain randomness, so the true rotation don't fit the model perfectly, and the maximum of likelyhood is not attained at the real rotations. We see now a clearer convergence until the end, the optimization continue

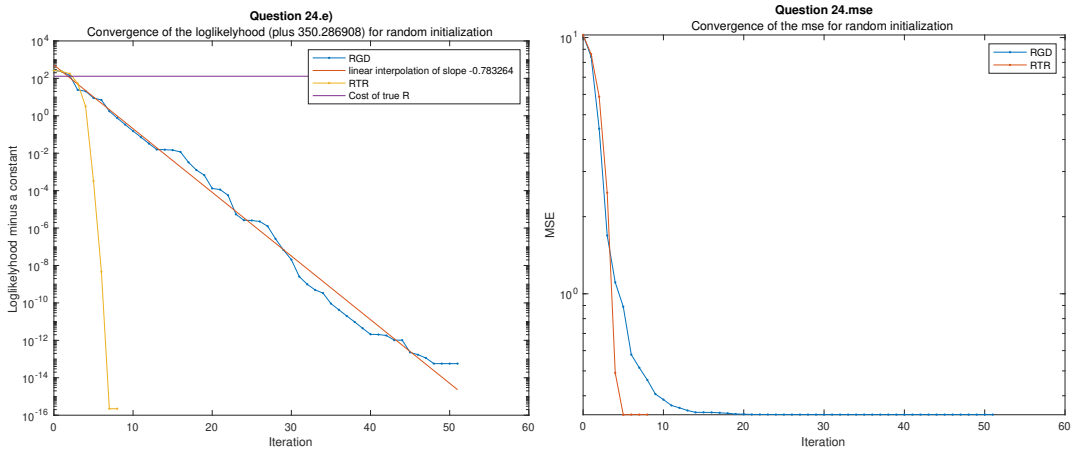


Figure 3: Question 24 d),e)

to make gain and is not staging. It seem that RGD has almost linear convergence, maybe a bit more since we observe a little of convexity. Again RGM is way better per iteration. The MSE error quickly decrease to zero, meaning that we recover the true rotations perfectly, even with the low noise.

f) We use the spectral initialization by writting `X0 = prob.init();`

g) We run RGD and RTR and get the ouput:

```
    Spectral initialization
RGD:
Last stepsize smaller than minimum allowed; options.minstepsize = 1e-10.
Total time is 0.130419 [s] (excludes statsfun)
RTR:
Gradient norm tolerance reached; options.tolgradnorm = 1e-07.
Total time is 0.234462 [s] (excludes statsfun)
```

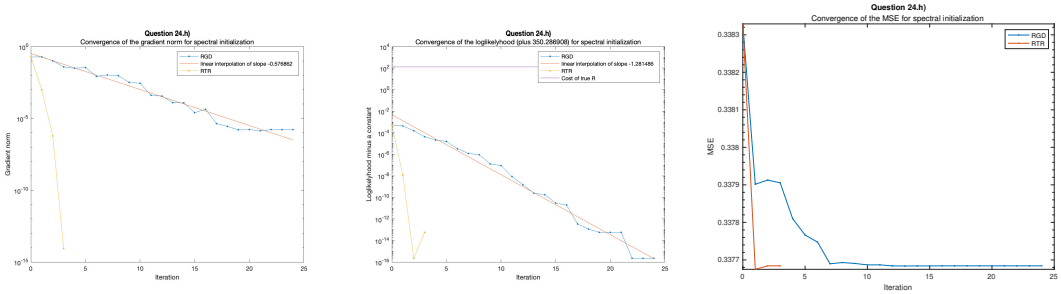


Figure 4: Question 24 h)

h) We plot the gradient norm and the objective function in Figure [4] with a log scale in the abscissa for the gradient. We see that we get a similar convergence as before, with again linear convergence for RGD with a rate however a bit smaller, and super-linear convergence for RTR. This could be explained from the fact that the initialization is yet better and put the steps on a nearer zone where the slope is smaller. We also note when the number of iterations stay the same, the running time of RTR has been divided by two, showing the interest of the spectral initialization for an algorithm that can actually reach the tolerance.

i) The code is tested by taking of the condition of identity of the first anchor. Everything run and converge so the test is positive.

**25.**

Here, we re-do the same experiments as in Question 24 but with some outliers, and for a larger problem. We plot the same graphs related to the points a)-h) in Figure [5] and Figure [6].

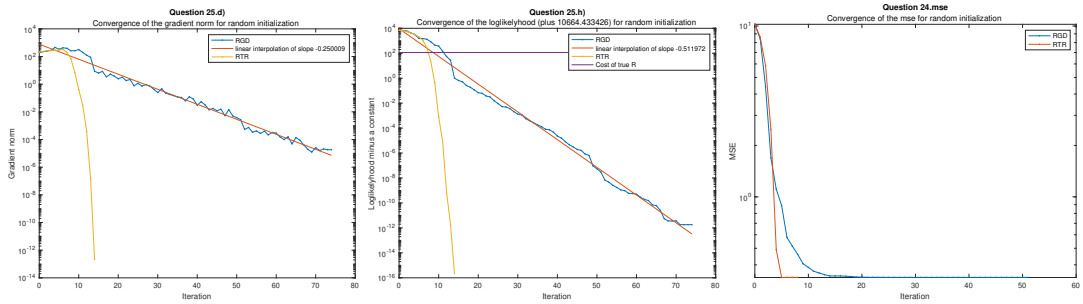


Figure 5: Question 25 d),e)

**26.**

Here, we re-do the same experiments as in Question 24 a),f)-h), with the data of Question 24 but for a incomplete Erdős-Rényi graph of density 0.75. We plot the same graphs related to point h) in Figure [7].

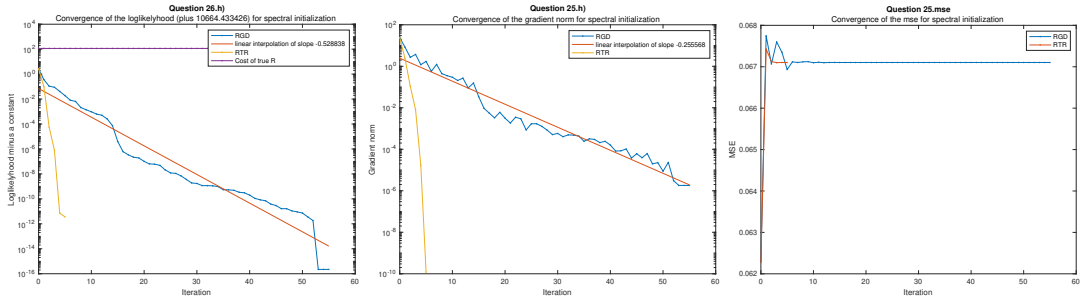


Figure 6: Question 25 h)

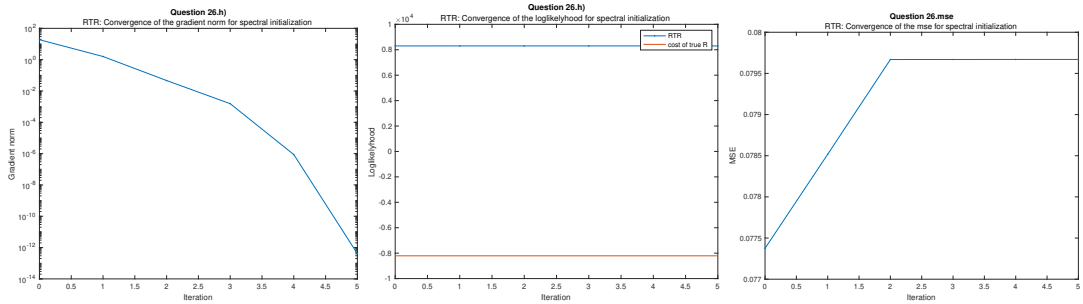


Figure 7: Question 26

**27.**

The spectral initialization has been thought while trying to optimize the objective function. We made the supposition that the density was not composite, that we have small noise but no outliers. Take the convention $\kappa_1 > \kappa_2$ such that $\kappa_1$ quantify the noise and $\kappa_2$ the outliers. We expect the spectral initialization to be good for $q = 1$ and $\kappa_1 > 0$, or at least for parameters such that $(q-1)\kappa_2$ small enough. As we make $q$ smaller, we make more outliers and don't know how the spectral initialization will behave. However, as $q$ goes smaller, the number of outlier augment and we would need a good initialization.

The purpose of this initialization is to avoid falling into some local non optimal minimum. The objective function is non convex, it means that some of the terms are not convex, and that by increasing the number of terms we can expect the number of local minima to increase. This motivates us to use a complete graph to accentuate the effect of the spectral initialization.

The dimension is kept to 3 by the motivation for the 3D space world, and to increase the effect of $q$, we set $\kappa_1 = 100$.

We decide to measure the MSE of different estimators, since it is the final measure that we actually care about for the final goal. We plot the MSE of a total random estimator, the MSE when we optimize it with RTR, and the MSE of the spectral initialization optimized with RTR. For the randomness of the initial point, we sample and plot a confidence interval with $alpha = 0.05$ risk, see Figure [8].

- We see that the optimization is clearly better than the random parameter, but that for low value of $q$, the estimator is bad, and even worse than a random initialization.

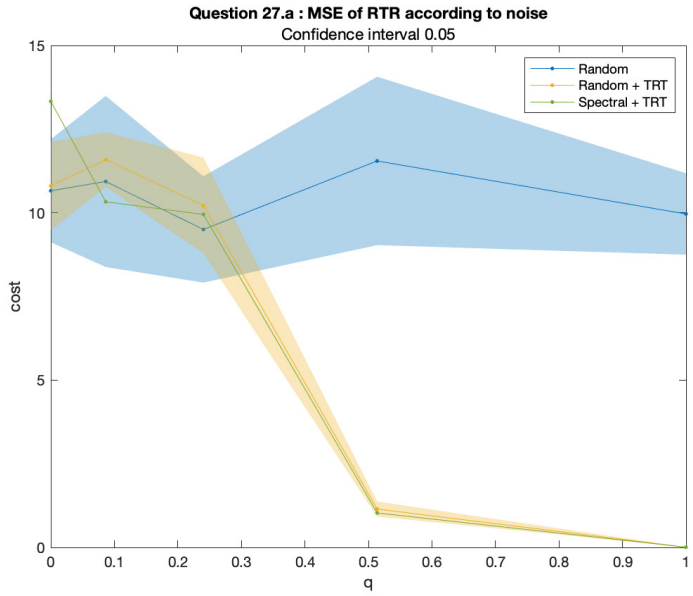


Figure 8: Question 27

# References

[1] N. Boumal et al. "Manopt, a Matlab Toolbox for Optimization on Manifolds". In: *Journal of Machine Learning Research* 15.42 (2014), pp. 1455–1459. URL: https://www.manopt.org.

[2] Nicolas Boumal. *An introduction to optimization on smooth manifolds*. Cambridge University Press, 2023. DOI: 10.1017/9781009166164. URL: https://www.nicolasboumal.net/book.

[3] Benoît Müller and Thomas Renard. *opti-manifolds-rotations-estimation*. 2023. URL: https://github.com/Benoit-Muller/opti-manifolds-rotations-estimation.