# Minimization on the Lie Group

Di Wang

Nov. 1st, 2018

Title **borrowed** from Taylor, C.J. and Kriegman, D.J., 1994. Minimization on the Lie group SO (3) and related manifolds. *Yale University*, *16*, p.155.

# Why Lie Group/Algebra?

- It is mathematically elegant.
- It associates 2D/3D rotation with 2D/3D Euclidean space.
- The gradient/Hessian involved are in simple formation.
- It is mainly applied in rotation-involved optimization.

➢ Before proceed, four rotation representation is recapped.

# Rotation Matrix

Representation: $$\mathbf{R} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \in \mathbb{R}^{3\times 3}$$

$$s.t.\ \mathbf{R}\mathbf{R}^T = 1,\ \det(\mathbf{R}) = 1$$

$$\mathbf{R} \approx \mathbf{I}$$

When rotation is small, i.e. rotation matrix is near identity matrix.

# Axis Angle

Representation: $\mathbf{u} = \begin{bmatrix} u_1 & u_2 & u_3 \end{bmatrix}^T \in \mathbb{R}^3$

$$Unit\ Rotation\ axis: \mathbf{a} = \frac{\mathbf{u}}{\|\mathbf{u}\|}$$

$$Rotation\ angle: \phi = \|\mathbf{u}\|$$

$$\mathbf{R}(\mathbf{u}) = \cos\phi \mathbf{I} + (1 - \cos\phi)\mathbf{a}\mathbf{a}^T - \sin\phi \mathbf{a}^\wedge$$

$\mathbf{R}(\mathbf{u}) \approx \mathbf{I} - (\mathbf{u})^\wedge$ When rotation is small, i.e. Fai is near zero.

$$\phi^\wedge = \begin{bmatrix} \phi_1 \\ \phi_2 \\ \phi_3 \end{bmatrix}^\wedge = \begin{bmatrix} 0 & -\phi_3 & \phi_2 \\ \phi_3 & 0 & -\phi_1 \\ -\phi_2 & \phi_1 & 0 \end{bmatrix} \in \mathbb{R}^{3\times 3}, \quad \phi \in \mathbb{R}^3.$$

Skew matrix or skew operator.

# Euler Angle

Representation: $\boldsymbol{\theta} = \begin{bmatrix} \theta_1 & \theta_2 & \theta_3 \end{bmatrix}^T \in \mathbb{R}^3$

$$\mathbf{C}_3 = \begin{bmatrix} \cos\theta_3 & \sin\theta_3 & 0 \\ -\sin\theta_3 & \cos\theta_3 & 0 \\ 0 & 0 & 1 \end{bmatrix}. \qquad \mathbf{C}_2 = \begin{bmatrix} \cos\theta_2 & 0 & -\sin\theta_2 \\ 0 & 1 & 0 \\ \sin\theta_2 & 0 & \cos\theta_2 \end{bmatrix}. \qquad \mathbf{C}_1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta_1 & \sin\theta_1 \\ 0 & -\sin\theta_1 & \cos\theta_1 \end{bmatrix}.$$

$$\begin{aligned} \mathbf{C}_{21}(\theta_3, \theta_2, \theta_1) &= \mathbf{C}_3(\theta_3)\mathbf{C}_2(\theta_2)\mathbf{C}_1(\theta_1) \\ &= \begin{bmatrix} c_2 c_3 & c_1 s_3 + s_1 s_2 c_3 & s_1 s_3 - c_1 s_2 c_3 \\ -c_2 s_3 & c_1 c_3 - s_1 s_2 s_3 & s_1 c_3 + c_1 s_2 s_3 \\ s_2 & -s_1 c_2 & c_1 c_2 \end{bmatrix}, \end{aligned}$$

$$\mathbf{R}(\boldsymbol{\theta}) \approx \mathbf{I} - \boldsymbol{\theta}^{\wedge}$$

When rotation is small, i.e. sin(x) = x, cos(x) = 1.0, and sin(x1)sin(x2) = 0.

# Unit Quaternion

Representation: $\mathbf{q} = \begin{bmatrix} q_x & q_y & q_z & q_w \end{bmatrix}^T \in \mathbb{R}^4$

$$s.t.\ q_x^2 + q_y^2 + q_z^2 + q_w^2 = 1$$

$$\mathbf{R}(\mathbf{q}) = \begin{pmatrix} 1 - 2q_y^2 - 2q_z^2 & 2q_x q_y - 2q_z \hat{q}_w & 2q_x q_z + 2q_y \hat{q}_w \\ 2q_x q_y + 2q_z \hat{q}_w & 1 - 2q_x^2 - 2q_z^2 & 2q_y q_z - 2q_x \hat{q}_w \\ 2q_x q_z - 2q_y \hat{q}_w & 2q_y q_z + 2q_x \hat{q}_w & 1 - 2q_x^2 - 2q_y^2, \end{pmatrix}$$

$$\mathbf{R}(\mathbf{q}) \approx \mathbf{I} + 2\mathbf{q}_{xyz}^{\wedge}$$

When rotation is small, i.e. qw = 1.0, qx = qy = qz = 0.0

# Interesting….But how do I calculate?

- MATLAB or Eigen!!!
- Build-in functions like quat2rotm(), axang2rotm(), eul2rotm(), or quat2axang().
- You may find that the resulting eul2rotm() or axang2rotm() is **the inverse** aforementioned rotation equation….
- That's fine, it is due to two conventions: someone likes to **rotate the point**, and someone likes to **rotate the coordinate frame**.
- Be consistent with one of them!

# Lie Group: SO(3) & SE(3)

$$SO(3) = \{\mathbf{C} \in \mathbb{R}^{3\times3} \mid \mathbf{C}\mathbf{C}^T = \mathbf{1}, \det \mathbf{C} = 1\}.$$

$$SE(3) = \left\{ \mathbf{T} = \begin{bmatrix} \mathbf{C} & \mathbf{r} \\ \mathbf{0}^T & 1 \end{bmatrix} \in \mathbb{R}^{4\times4} \,\middle|\, \mathbf{C} \in SO(3),\, \mathbf{r} \in \mathbb{R}^3 \right\}.$$

| property | $SO(3)$ | $SE(3)$ |
|---|---|---|
| closure | $\mathbf{C}_1, \mathbf{C}_2 \in SO(3)$ $\Rightarrow \mathbf{C}_1\mathbf{C}_2 \in SO(3)$ | $\mathbf{T}_1, \mathbf{T}_2 \in SE(3)$ $\Rightarrow \mathbf{T}_1\mathbf{T}_2 \in SE(3)$ |
| associativity | $\mathbf{C}_1(\mathbf{C}_2\mathbf{C}_3) = (\mathbf{C}_1\mathbf{C}_2)\mathbf{C}_3$ $= \mathbf{C}_1\mathbf{C}_2\mathbf{C}_3$ | $\mathbf{T}_1(\mathbf{T}_2\mathbf{T}_3) = (\mathbf{T}_1\mathbf{T}_2)\mathbf{T}_3$ $= \mathbf{T}_1\mathbf{T}_2\mathbf{T}_3$ |
| identity | $\mathbf{C}, \mathbf{1} \in SO(3)$ $\Rightarrow \mathbf{C}\mathbf{1} = \mathbf{1}\mathbf{C} = \mathbf{C}$ | $\mathbf{T}, \mathbf{1} \in SE(3)$ $\Rightarrow \mathbf{T}\mathbf{1} = \mathbf{1}\mathbf{T} = \mathbf{T}$ |
| invertibility | $\mathbf{C} \in SO(3)$ $\Rightarrow \mathbf{C}^{-1} \in SO(3)$ | $\mathbf{T} \in SE(3)$ $\Rightarrow \mathbf{T}^{-1} \in SE(3)$ |

# Lie Algebra: so(3)

$$
\begin{aligned}
\text{vectorspace:} \quad & \mathfrak{so}(3) = \{\boldsymbol{\Phi} = \phi^\wedge \in \mathbb{R}^{3\times 3} \mid \phi \in \mathbb{R}^3\}, \\
\text{field:} \quad & \mathbb{R}, \\
\text{Lie bracket:} \quad & [\boldsymbol{\Phi}_1, \boldsymbol{\Phi}_2] = \boldsymbol{\Phi}_1\boldsymbol{\Phi}_2 - \boldsymbol{\Phi}_2\boldsymbol{\Phi}_1,
\end{aligned}
$$

Used in BCH formula.

$$
\phi^\wedge = \begin{bmatrix} \phi_1 \\ \phi_2 \\ \phi_3 \end{bmatrix}^\wedge = \begin{bmatrix} 0 & -\phi_3 & \phi_2 \\ \phi_3 & 0 & -\phi_1 \\ -\phi_2 & \phi_1 & 0 \end{bmatrix} \in \mathbb{R}^{3\times 3}, \quad \phi \in \mathbb{R}^3.
$$

$$
\mathit{Cross\ product}: \ \mathbf{a} \times \mathbf{b} = \mathbf{a}^\wedge \mathbf{b}
$$

# Relationship between SO(3) & so(3)

$$\mathbb{R}^3 \rightarrow SO(3): \ \exp(\phi^\wedge) = \mathbf{I} + \frac{\sin(\|\phi\|)}{\|\phi\|}\phi^\wedge + \frac{1-\cos(\|\phi\|)}{\|\phi\|^2}(\phi^\wedge)^2 . \qquad \exp(\phi^\wedge) \approx \mathbf{I} + \phi^\wedge$$

$$SO(3) \rightarrow \mathbb{R}^3: \ \log(\mathrm{R}) = \frac{\varphi \cdot (\mathrm{R} - \mathrm{R}^\mathsf{T})}{2\sin(\varphi)} \text{ with } \varphi = \cos^{-1}\left(\frac{\mathrm{tr}(\mathrm{R}) - 1}{2}\right)$$

Exp() is often employed for rotation perturbation, i.e. a rotation is perturbed by a very small rotation.
Log() is often employed as distance metric for measuring the error:

$$Err(\mathbf{R}_1, \mathbf{R}_2) = \log(\mathbf{R}_1^{-1}\mathbf{R}_2)$$

# BCH Formula

$$If \;\; \exp(A)\exp(B) = \exp(C), \;\; then\ is\ C = A + B?$$

$$Nope!Use\ BCH\ \ fomula:$$

$$\begin{aligned}\ln(\exp(\mathbf{A})\exp(\mathbf{B})) = \mathbf{A} + \mathbf{B} + \frac{1}{2}[\mathbf{A},\mathbf{B}] \\ + \frac{1}{12}[\mathbf{A},[\mathbf{A},\mathbf{B}]] - \frac{1}{12}[\mathbf{B},[\mathbf{A},\mathbf{B}]] - \frac{1}{24}[\mathbf{B},[\mathbf{A},[\mathbf{A},\mathbf{B}]]] \\ - \frac{1}{720}([[[[\mathbf{A},\mathbf{B}],\mathbf{B}],\mathbf{B}],\mathbf{B}] + [[[[\mathbf{B},\mathbf{A}],\mathbf{A}],\mathbf{A}],\mathbf{A}]) \\ + \frac{1}{360}([[[[\mathbf{A},\mathbf{B}],\mathbf{B}],\mathbf{B}],\mathbf{A}] + [[[[\mathbf{B},\mathbf{A}],\mathbf{A}],\mathbf{A}],\mathbf{B}]) \\ + \frac{1}{120}([[[[\mathbf{A},\mathbf{B}],\mathbf{A}],\mathbf{B}],\mathbf{A}] + [[[[\mathbf{B},\mathbf{A}],\mathbf{B}],\mathbf{A}],\mathbf{B}]) + \cdots .\end{aligned}$$

BCH formula is widely used in visual inertial odometry for autonomous drones.

$$\mathrm{Exp}(\phi + \delta\phi) \approx \mathrm{Exp}(\phi)\,\mathrm{Exp}(\mathrm{J}_r(\phi)\delta\phi).$$

$$\mathrm{J}_r(\phi) = \mathbf{I} - \frac{1-\cos(\|\phi\|)}{\|\phi\|^2}\phi^\wedge + \frac{\|\phi\| - \sin(\|\phi\|)}{\|\phi^3\|}(\phi^\wedge)^2.$$

# How to Use Lie Algebra in Optimization?

$$\mathbf{x}^* = \arg\min_{\mathbf{x}} f(\mathbf{x}), \qquad \mathbf{x} \in SO(3)$$

$$\mathbf{x}_{new} = \mathbf{x}_{old} \boxplus \alpha \left.\frac{\partial f}{\partial \mathbf{x}}\right|_{\mathbf{x}=\mathbf{x}_{old}}$$ Gradient descent on Lie group.

$$\exp(\boldsymbol{\varepsilon}) \approx \mathbf{I} + \boldsymbol{\varepsilon}^{\wedge}$$

$$\mathbf{a}^{\wedge}\mathbf{b} = -\mathbf{b}^{\wedge}\mathbf{a}$$

**Useful equations**

$$SO(3) \times \mathbb{R}^3 \to SO(3): \quad \mathbf{x} \boxplus \boldsymbol{\varepsilon} = \mathrm{Exp}(\boldsymbol{\varepsilon})\mathbf{R}$$

**Left perturbation scheme.**

$$\frac{\partial f}{\partial \mathbf{x}} = \lim_{\boldsymbol{\varepsilon}\to 0} \frac{f(\mathbf{x} \boxplus \boldsymbol{\varepsilon}) - f(\mathbf{x})}{\boldsymbol{\varepsilon}} = \lim_{\boldsymbol{\varepsilon}\to 0} \frac{f\left(\left(\mathbf{I} + \boldsymbol{\varepsilon}^{\wedge}\right)\mathbf{R}\right) - f(\mathbf{R})}{\boldsymbol{\varepsilon}}$$

$$\textit{When } \mathbf{f} = \mathbf{Rv}:$$

In practice, gradient/Jacobian deduction are error-prone, **remember to verify them in numerical way.**

Jacobian matrix:

$$\frac{\partial \mathbf{Rv}}{\partial \mathbf{x}} = \lim_{\boldsymbol{\varepsilon}\to 0} \frac{\left(\mathbf{I} + \boldsymbol{\varepsilon}^{\wedge}\right)\mathbf{Rv} - \mathbf{Rv}}{\boldsymbol{\varepsilon}} = \lim_{\boldsymbol{\varepsilon}\to 0} \frac{\boldsymbol{\varepsilon}^{\wedge}\mathbf{Rv}}{\boldsymbol{\varepsilon}} = -(\mathbf{Rv})^{\wedge}$$

# More Generalize Case

$$f(\mathbf{x}) = \sum_{i=1}^{N} \left(\mathbf{R}\mathbf{q}_i - \mathbf{p}_i\right)^T \boldsymbol{\Sigma} \left(\mathbf{R}\mathbf{q}_i - \mathbf{p}_i\right)$$

$$\mathbf{R}_{new} \to \mathrm{Exp}(\boldsymbol{\varepsilon})\mathbf{R} \approx \left(\mathbf{I} + \boldsymbol{\varepsilon}^{\wedge}\right)\mathbf{R}$$

$$\begin{aligned} f_{new}(\mathbf{x}) = g(\boldsymbol{\varepsilon}) &= \sum_{i=1}^{N} \left(\mathbf{R}\mathbf{q}_i - \mathbf{p}_i + \boldsymbol{\varepsilon}^{\wedge}\mathbf{R}\mathbf{q}_i\right)^T \boldsymbol{\Sigma} \left(\mathbf{R}\mathbf{q}_i - \mathbf{p}_i + +\boldsymbol{\varepsilon}^{\wedge}\mathbf{R}\mathbf{q}_i\right) \\ &= \sum_{i=1}^{N} \left(\mathbf{v}_i - \left(\mathbf{R}\mathbf{q}_i\right)^{\wedge} \boldsymbol{\varepsilon}\right)^T \boldsymbol{\Sigma} \left(\mathbf{v}_i - \left(\mathbf{R}\mathbf{q}_i\right)^{\wedge} \boldsymbol{\varepsilon}\right) \\ &= \boldsymbol{\varepsilon}\mathbf{H}\boldsymbol{\varepsilon} + 2\mathbf{b}^T\boldsymbol{\varepsilon} + const \end{aligned}$$

$$\boldsymbol{\varepsilon}^{opt} = \arg\min_{\boldsymbol{\varepsilon}} \left(\boldsymbol{\varepsilon}\mathbf{H}\boldsymbol{\varepsilon} + 2\mathbf{b}^T\boldsymbol{\varepsilon}\right) = -\mathbf{H}^{-1}\mathbf{b}$$

$$\mathbf{R}_{new} = Exp\left(\boldsymbol{\varepsilon}^{opt}\right)\mathbf{R}$$

# Plus Operator in g2o

$$\mathbf{x} = (\mathbf{t}, \mathbf{q}) \in SE(3)$$

$$s.t.\ \|\mathbf{q}\| = 1$$

$$SE(3) \times \mathbb{R}^7 \to SE(3) : \mathbf{x} \boxplus \boldsymbol{\varepsilon} = \mathrm{ToISO3D}(\mathbf{x})\,\mathrm{ToISO3D}(\boldsymbol{\varepsilon}) = \left(\mathbf{R}\mathbf{t}_\varepsilon + \mathbf{t}, \mathrm{rotToQuat}\left(\mathbf{R}\mathbf{R}_\varepsilon\right)\right) \in SE(3)$$

$$ToISO3D(\mathbf{x}) = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix},\ \mathbf{R} = \mathrm{quatToRot}(\mathbf{q})$$

A possible plus operator:

$$\mathbf{x} = (\mathbf{t}, \mathbf{u}) \in SE(3)$$

$$SE(3) \times \mathbb{R}^6 \to SE(3) : \mathbf{x} \boxplus \boldsymbol{\varepsilon} = \left(\mathbf{t} + \mathbf{t}_\varepsilon, \mathrm{rotToAxang}\left(\mathbf{R}_\varepsilon \mathbf{R}\right)\right) \in SE(3)$$

# The-state-of-the-art Algorithms

- What kind of rotation are employed in popular algorithms?
  - Point-to-point ICP: Rotation matrix.
  - Point-to-plane ICP: Euler angle.
  - Plane-to-plane ICP(GICP): Euler angle.
  - NDT: Euler angle.
  - g2o: Quaternion.
  - Visual Inertial Odometry (VIO): Axis-angle/Lie algebra.
- It seems that **Lie algebra is popular in computer vision community.** Quaternion is a viable choice for **robotics**.

# Conclusion

- Remember the Exp() and Log() operator in Lie group and algebra.
- In practice, the Exp() and Log() operators are often replaced by more simple plus and minus operators.