



# A Tutorial on Graph-Based SLAM

---

# はじめに

---

- SLAMへのアプローチ

- フィルタリングアプローチ (オンラインSLAMと呼ばれる)

システムの状態が現在のロボットの位置と地図で構成されるオンライン状態推定として問題をモデル化する

例) カルマン情報フィルタ

パーティクルフィルタ

- 平滑化アプローチ

測定した情報セットからロボットの完全な軌道を推定する。これらの手法は **最小二乗問題**として扱われる。



# はじめに

---

- グラフベースSLAM
  - 頂点: ロボットの姿勢・ランドマーク
  - 辺 : センサデータ

しかし、センサデータは誤差を含む

⇒ **誤差最小化問題**となる



# SLAMの定式化

---

$$p(x_{1:T}, m | z_{1:T}, u_{1:T}, x_0)$$

$x_{1:t} = \{x_1, \dots, x_t\}$ : ロボットの軌道

$u_{1:t} = \{u_1, \dots, u_t\}$  オドメトリ

$z_{1:t} = \{z_1, \dots, z_t\}$ : 環境情報

$m$ : 地図

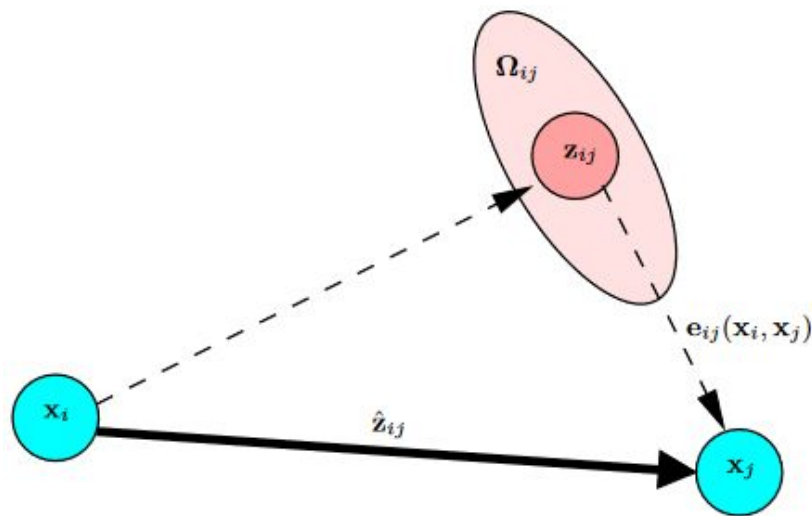
オドメトリ・環境情報・現在地が与えられたときの軌道と地図を推定する



# グラフベースSLAM

---

# グラフのエッジを定義するための関数とその量



$x_i$  と  $x_j$  を接続する図

$x = (x_1, \dots, x_T)$  : ロボットの姿勢

$z_{ij}, \Omega_{ij}$  : ノード  $i, j$  を接続するための仮定の測定値の入った情報行列

$\hat{z}_{ij}(x_i, x_j)$  : ノード  $x_i, x_j$  から測定値を予測する

$e(x_i, x_j, z_{ij})$  誤差関数



# 目的関数

$$\mathbf{F}(\mathbf{x}) = \sum_{\langle i,j \rangle \in \mathcal{C}} \underbrace{\mathbf{e}_{ij}^T \boldsymbol{\Omega}_{ij} \mathbf{e}_{ij}}_{\mathbf{F}_{ij}},$$

$$\mathbf{x}^* = \underset{\mathbf{x}}{\operatorname{argmin}} \mathbf{F}(\mathbf{x}).$$

- 目的
  - 観測値の負の対数尤度  $\mathbf{F}(\mathbf{x})$  を最小にするノード $\mathbf{x}^*$ の構成を見つけることである
- 手法
  - 反復的局所線形化による誤差最小化 を行う  
例) ガウス・ニュートン法  
Levenberg-Marquardtアルゴリズム  
※ロボットの姿勢の初期推定値  $\check{\mathbf{x}}$   
既知である必要がある



# 反復的局所線形化による誤差最小化 ( 1 / 5 )

---

現在の初期推定値  $\check{\mathbf{x}}$  を中心とした一次テイラー展開によって誤差関数を近似する

$$\mathbf{e}_{ij}(\check{\mathbf{x}}_i + \Delta \mathbf{x}_i, \check{\mathbf{x}}_j + \Delta \mathbf{x}_j) = \mathbf{e}_{ij}(\check{\mathbf{x}} + \Delta \mathbf{x}) \quad (6)$$

$$\simeq \mathbf{e}_{ij} + \mathbf{J}_{ij} \Delta \mathbf{x}. \quad (7)$$






## 反復的局所線形化による誤差最小化 ( 2 / 5 )

(7)の式をF(x)の式に代入する

$$\mathbf{F}_{ij}(\check{\mathbf{x}} + \Delta \mathbf{x})$$
$$= \mathbf{e}_{ij}(\check{\mathbf{x}} + \Delta \mathbf{x})^T \boldsymbol{\Omega}_{ij} \mathbf{e}_{ij}(\check{\mathbf{x}} + \Delta \mathbf{x}) \quad (8)$$

$$\simeq (\mathbf{e}_{ij} + \mathbf{J}_{ij} \Delta \mathbf{x})^T \boldsymbol{\Omega}_{ij} (\mathbf{e}_{ij} + \mathbf{J}_{ij} \Delta \mathbf{x}) \quad (9)$$

$$= \underbrace{\mathbf{e}_{ij}^T \boldsymbol{\Omega}_{ij} \mathbf{e}_{ij}}_{c_{ij}} + 2 \underbrace{\mathbf{e}_{ij}^T \boldsymbol{\Omega}_{ij} \mathbf{J}_{ij}}_{\mathbf{b}_{ij}} \Delta \mathbf{x} + \Delta \mathbf{x}^T \underbrace{\mathbf{J}_{ij}^T \boldsymbol{\Omega}_{ij} \mathbf{J}_{ij}}_{\mathbf{H}_{ij}} \Delta \mathbf{x} \quad (10)$$

$$= c_{ij} + 2\mathbf{b}_{ij} \Delta \mathbf{x} + \Delta \mathbf{x}^T \mathbf{H}_{ij} \Delta \mathbf{x} \quad (11)$$


## 反復的局所線形化による誤差最小化 ( 3 / 5 )

---

局所近似により次のように書き換えることが出来る。

そして、 $c = \sum c_{ij}, b = \sum b_{ij}, H = \sum H_{ij}$  することで (13)  $\Rightarrow$  (14) と置き換わる

$$\mathbf{F}(\check{\mathbf{x}} + \Delta \mathbf{x}) = \sum_{\langle i, j \rangle \in \mathcal{C}} \mathbf{F}_{ij}(\check{\mathbf{x}} + \Delta \mathbf{x}) \quad (12)$$

$$\simeq \sum_{\langle i, j \rangle \in \mathcal{C}} c_{ij} + 2\mathbf{b}_{ij} \Delta \mathbf{x} + \Delta \mathbf{x}^T \mathbf{H}_{ij} \Delta \mathbf{x} \quad (13)$$

$$= c + 2\mathbf{b}^T \Delta \mathbf{x} + \Delta \mathbf{x}^T \mathbf{H} \Delta \mathbf{x}. \quad (14)$$



# 反復的局所線形化による誤差最小化（ 4 / 5 ）

---

$\Lambda x$ を最小化することで解くことが出来る

$$\mathbf{H} \Delta \mathbf{x}^* = -\mathbf{b}. \quad (15)$$



# 反復的局所線形化による誤差最小化（ 5 / 5 ）

---

行列Hは測定誤差をヤコビ行列を通して軌道の空間に投影することが出来る。構造上、Hは疎な行列であり、制約によって接続されたポーズには0がない。非ゼロのブロック数は制約の数の2倍である。これにより式 (15)を**Sparse Cholesky**により解くことが出来る。線形化された解は、計算された増分を初期推測に加えることで得られる。

$$\mathbf{x}^* = \check{\mathbf{x}} + \Delta \mathbf{x}^*. \quad (16)$$

一般的なガウス・ニュートン法は (14)の線形化、(15)の解、(16)の更新の繰り返しである。各反復では前回の結果が用いられる。上記の手順は一般的なアプローチであり、パラメータ  $\mathbf{x}$  がユークリッドであることを前提としている。そのため、SLAMでは有効でない可能性もある。



# 線形化されたシステムへの考察 ( 1 / 3 )

式(14)から、行列Hとベクトルbは、制約ごとに1つずつの行列とベクトルを合計することで得られる。全ての制約条件はシステムに負荷項をもたらす。この加算による構造は誤差関数のヤコビ行列に依存する。制約の誤差関数は 2つのノードの値にのみ依存するため、(7)のヤコビ行列は次のような形になる。

$$\mathbf{J}_{ij} = \begin{pmatrix} 0 \cdots 0 & \underbrace{\mathbf{A}_{ij}}_{\text{node } i} & 0 \cdots 0 & \underbrace{\mathbf{B}_{ij}}_{\text{node } j} & 0 \cdots 0 \end{pmatrix}. \quad (17)$$



## 線形化されたシステムへの考察（ 2 / 3 ）

ここで、 $A_{ij}$ と $B_{ij}$ は、 $x_i$ と $x_j$ に対する誤差関数の微分である。式 (10)から、ブロック行列  $H_{ij}$ の構造は次のようになる。

(見やすさの為に0は省略している。)

$$\mathbf{H}_{ij} = \begin{pmatrix} \ddots & & & & \\ & \mathbf{A}_{ij}^T \boldsymbol{\Omega}_{ij} \mathbf{A}_{ij} & \cdots & \mathbf{A}_{ij}^T \boldsymbol{\Omega}_{ij} \mathbf{B}_{ij} & \\ & \vdots & \ddots & \vdots & \\ & \mathbf{B}_{ij}^T \boldsymbol{\Omega}_{ij} \mathbf{A}_{ij} & \cdots & \mathbf{B}_{ij}^T \boldsymbol{\Omega}_{ij} \mathbf{B}_{ij} & \\ & & & \ddots & \end{pmatrix} \quad (18)$$

$$\mathbf{b}_{ij} = \begin{pmatrix} \vdots \\ \mathbf{A}_{ij}^T \boldsymbol{\Omega}_{ij} \mathbf{e}_{ij} \\ \vdots \\ \mathbf{B}_{ij}^T \boldsymbol{\Omega}_{ij} \mathbf{e}_{ij} \\ \vdots \end{pmatrix} \quad (19)$$



# 線形化されたシステムへの考察 ( 3 / 3 )

アルゴリズム 1は、ロボットの姿勢に関する事後情報の平均値と情報行列を決定する反復的なガウス・ニュートン法を要約したものである。システムのほとんどの構造は疎であるため、**メモリ効率の良い表現を使用したほうが良い**。ヘッセ行列の構造は、グラフの接続性から事前に知られているので、反復の最初に一度だけヘッセ行列を事前に割り当て、新たな線形化が必要なときに、すべてのエッジをループしてその場で更新したほうが良い。各エッジは、ブロック  $H_{[ii]}$ 、 $H_{[ij]}$ 、 $H_{[ji]}$ 、 $H_{[jj]}$ と、係数ベクトルのブロック  $b_{[i]}$ および  $b_{[j]}$ に寄与する。さらに、 $H$ の上三角部は対称的なので、上三角部のみを計算するという最適化も行われている。アルゴリズム 1では、一般性を損なわないように、最初のノード  $ex_1$ を固定する。ポーズ・グラフの特定のノードを固定する別の方法は、式 15の線形システムの  $k$ 番目のブロックの行と  $k$ 番目のブロックの列を抑制することである。

```
Require:  $\check{x} = \check{x}_{1:T}$ : 初期予測,  $\mathcal{C} = \{\langle e_{ij}(\cdot), \Omega_{ij} \rangle\}$ :  
制約  
Ensure:  $x^*$ : 解,  $H^*$ : 新たな情報行列  
// 尤度が最大の解を求める  
while  $\neg \text{converged}$  do  
   $b \leftarrow 0$    $H \leftarrow 0$   
  for all  $\langle e_{ij}, \Omega_{ij} \rangle \in \mathcal{C}$  do  
    // ヤコビ行列  $A_{ij}$ ,  $B_{ij}$  及び誤差関数を計算する  
     $A_{ij} \leftarrow \left. \frac{\partial e_{ij}(x)}{\partial x_i} \right|_{x=\check{x}}$    $B_{ij} \leftarrow \left. \frac{\partial e_{ij}(x)}{\partial x_j} \right|_{x=\check{x}}$   
    // この制約条件の線形システムへの寄与度を計算  
     $H_{[ii]} += A_{ij}^T \Omega_{ij} A_{ij}$    $H_{[ij]} += A_{ij}^T \Omega_{ij} B_{ij}$   
     $H_{[ji]} += B_{ij}^T \Omega_{ij} A_{ij}$    $H_{[jj]} += B_{ij}^T \Omega_{ij} B_{ij}$   
    // 係数ベクトルの算出  
     $b_{[i]} += A_{ij}^T \Omega_{ij} e_{ij}$    $b_{[j]} += B_{ij}^T \Omega_{ij} e_{ij}$   
  end for  
  // 最初のノードを固定する  
   $H_{[11]} += I$   
  // Sparse Cholesky を用いて線形システムを解く  
   $\Delta x \leftarrow \text{solve}(H \Delta x = -b)$   
  // パラメータを更新  
   $\check{x} += \Delta x$   
end while  
 $x^* \leftarrow \check{x}$   
 $H^* \leftarrow H$   
// 最初のノードを解放(?) release the first node  
 $H_{[11]} -= I$   
return  $\langle x^*, H^* \rangle$ 
```



# 多様体における最小二乗法（1 / ）

- SLAMの地図が扱う空間の要素

- 並進ベクトル
  - 回転成分
- ）  $\longrightarrow$  回転成分は非ユークリッド空間の存在である

## ➡ 多様体上で最適化を行う

多様体: 大局的には必ずしもユークリッドではないが、局所的にはユークリッドとみなすことができる数学的空間

基礎となる空間を多様体と考え、ユークリッド空間の局所的な変化 $\Delta x$ を多様体上の変化  $\Delta x \mapsto x \boxplus \Delta x$  にマッピングする演算子 $\boxplus$ を定義する。





# 多様体における最小二乗法 ( 1 / n )

- SLAMの地図が扱う空間の要素

- 並進ベクトル
  - 回転成分
- 回転成分は非ユークリッド空間の存在である

## → 多様体上で最適化を行う

多様体: 大局的には必ずしもユークリッドではないが、局所的にはユークリッドとみなすことができる数学的空間

基礎となる空間を多様体と考え、ユークリッド空間の局所的な変化 $\Delta x$ を多様体上の変化  $\Delta x \mapsto x \boxplus \Delta x$  にマッピングする演算子 $\boxplus$ を定義する。

