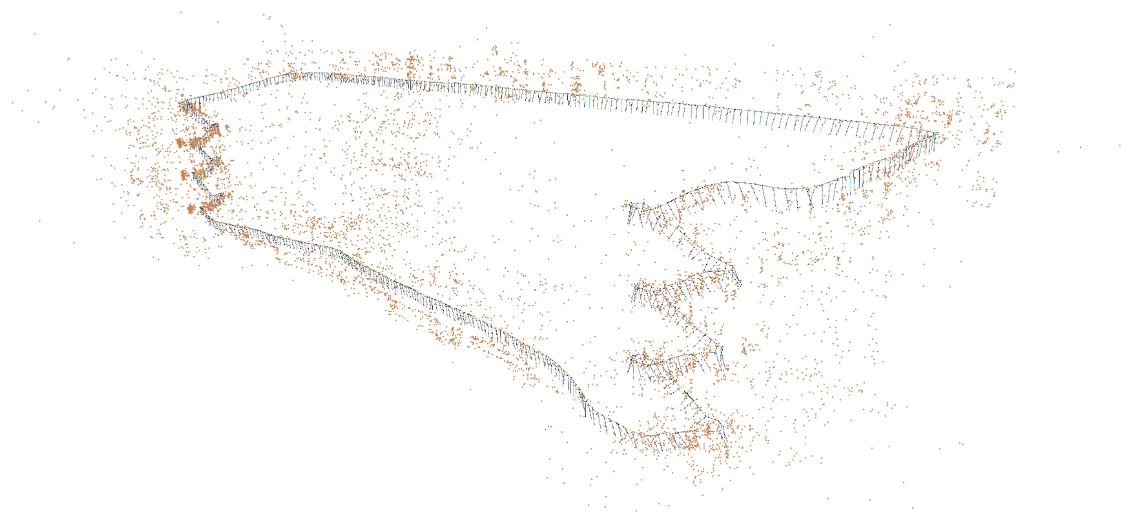
# Probabilistic Robotics Course

# Factor Graphs

**Giorgio Grisetti**
grisetti@diag.uniroma1.it

Department of Computer, Control, and Management Engineering
Sapienza University of Rome



# Large Problems

Many relevant estimation problems require estimating a large number of variables given a very large number of measurements.

Examples include

- Pose-graphs
- Pose-Landmark
- Calibration
- **<add your own>**
- **<combine the above problems>**

**The larger the problem, the slower the solution**

# Large and Sparse Problems

Despite the high number of variables, a single measurement is typically only affected by a subset of state variables

Examples:

- *Landmark measurement* determined by :
  - the observer position $X_r^{[n]}$
  - the position of the landmark $X_l^{[m]}$
- Pose-Pose measurement determined by
  - the observer position $X_r^{[i]}$
  - the observed position $X_r^{[j]}$

# Graphical Representation

Formally we can highlight the dependency of a measurement from a subset of state variables by the following notation

error function, for meas $z^{[k]}$

state variables

$$\mathbf{x}^T = (\mathbf{x}^{[1]T}, \ldots, \mathbf{x}^{[n]T})$$

$$\mathbf{e}^{[k]}(\mathbf{x}^{[k]})$$

$$\mathbf{x}^{[k]T} = (\mathbf{x}^{[k_1]T}, \ldots, \mathbf{x}^{[k_q]T}) \subset (\mathbf{x}^{[1]T}, \ldots, \mathbf{x}^{[n]T})$$

subset of vars affecting $z^{[k]}$

log likelihood

$$F(\mathbf{x}) = \sum_{k \in \mathcal{C}} \underbrace{\mathbf{e}^{[k]}(\mathbf{x}^{k})^T \mathbf{\Omega}^{[k]} \mathbf{e}^{[k]}(\mathbf{x}^{[k]})}_{F^{[k]}}$$

optimum

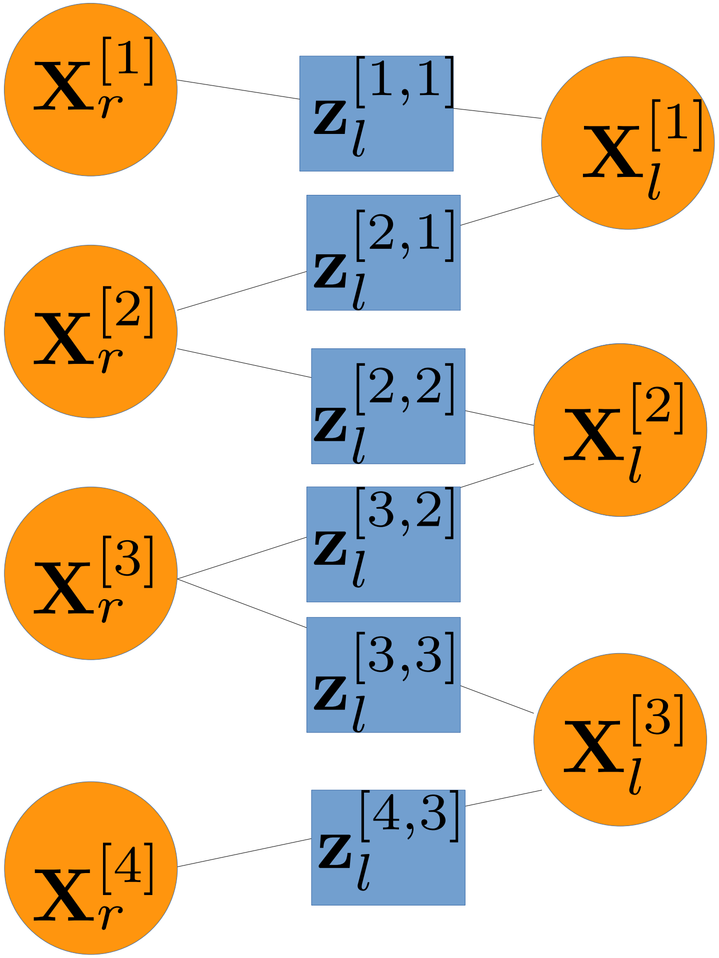
$$\mathbf{x}^* = \operatorname{argmin}_{\mathbf{x}} F(\mathbf{x})$$

# Graphical Representation

We can represent the problem with a graph

- A node for each state variable
- A node for each measurement
- An edge between a variable and a measurement if they are dependant

Example: pose-landmark SLAM

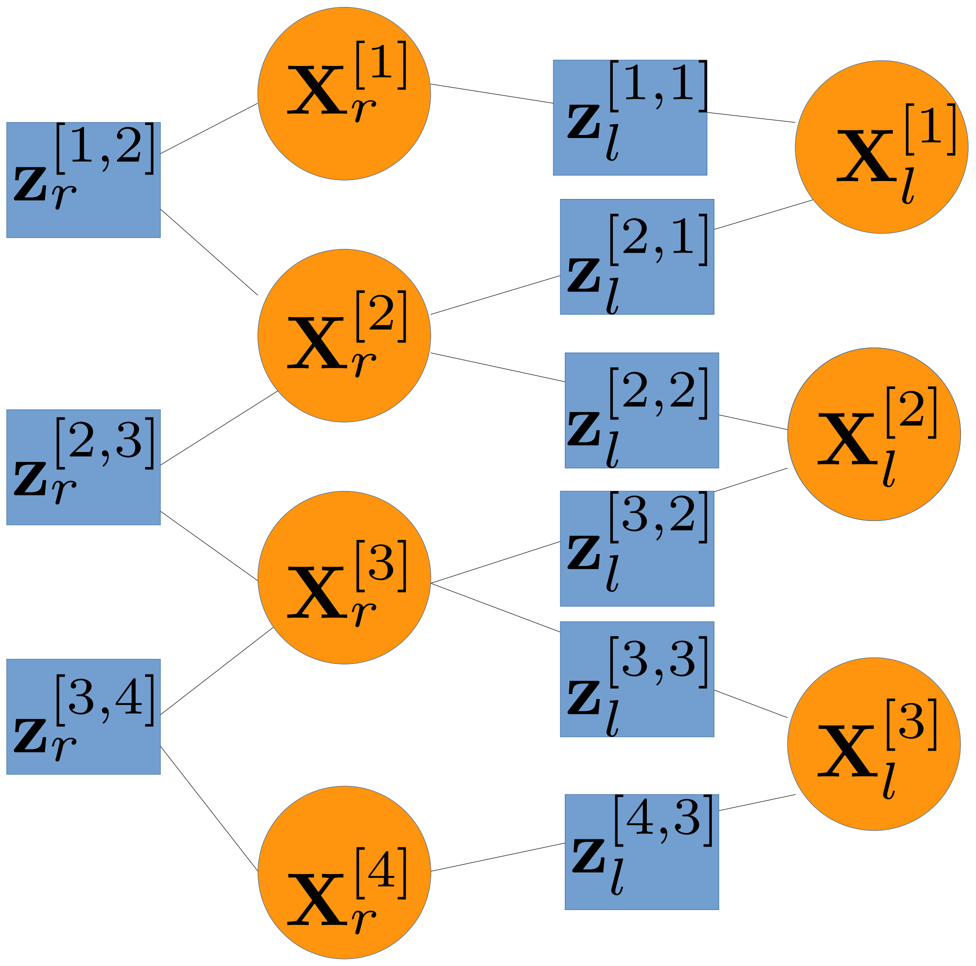


# Graphical Representation

We can represent the problem with a graph

- A node for each state variable
- A node for each measurement
- An edge between a variable and a measurement if they are dependant

Example: landmark+ odometry



# Graphical Representation

This representation is known as a factor graph.

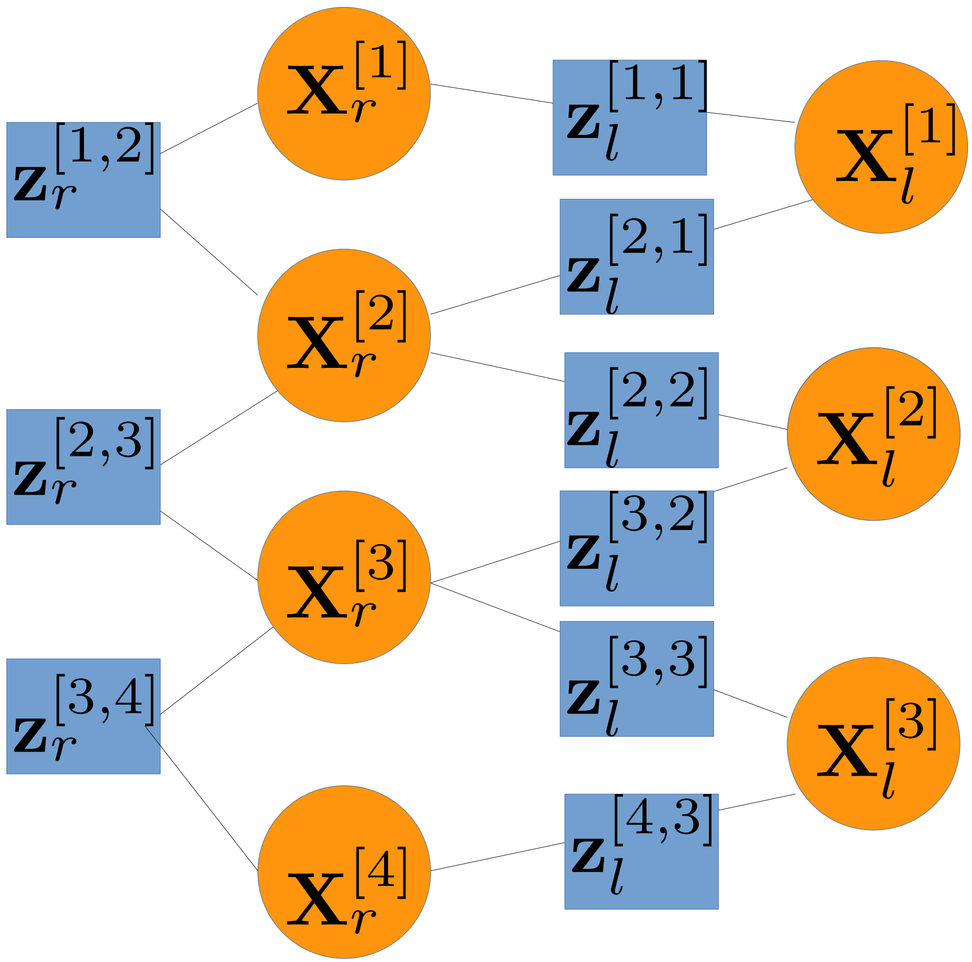
Can represent arbitrary factored functions

In our case we restrict to Gaussian Likelihoods

It is bipartite:

- two variable nodes are connected only through a measurement
- factors can be seen as hyper-edges connecting multiple variables

Example: landmark+ odometry



# Structure

The contribution to the **H** matrix for a measurement $z^{[k]}$ measurement will affect only the state components in $x^{[k]}$

$$\mathbf{J}^{[k]} = \begin{pmatrix}\mathbf{0} \cdots \mathbf{0}\; \mathbf{J}^{[k_1]} \;\cdots\; \mathbf{J}^{[k_i]} \;\cdots \mathbf{0} \;\cdots\; \mathbf{J}^{[k_q]} \mathbf{0} \cdots \mathbf{0}\end{pmatrix}$$

$$\mathbf{H}^{[k]} = \begin{pmatrix} \ddots & & & & & & \\ & \mathbf{J}^{[k_1]T}\boldsymbol{\Omega}^{[k]}\mathbf{J}^{[k_1]} & \cdots & \mathbf{J}^{[k_1]T}\boldsymbol{\Omega}^{[k]}\mathbf{J}^{[k_i]} & \cdots & \mathbf{J}^{[k_1]T}\boldsymbol{\Omega}^{[k]}\mathbf{J}^{[k_q]} & \\ & \vdots & & \vdots & & \vdots & \\ & \mathbf{J}^{[k_i]T}\boldsymbol{\Omega}^{[k]}\mathbf{J}^{[k_1]} & \cdots & \mathbf{J}^{[k_i]T}\boldsymbol{\Omega}^{[k]}\mathbf{J}^{[k_i]} & \cdots & \mathbf{J}^{[k_i]T}\boldsymbol{\Omega}^{[k]}\mathbf{J}^{[k_q]} & \\ & \vdots & & \vdots & & \vdots & \\ & \mathbf{J}^{[k_q]T}\boldsymbol{\Omega}^{[k]}\mathbf{J}^{[k_1]} & \cdots & \mathbf{J}^{[k_q]T}\boldsymbol{\Omega}^{[k]}\mathbf{J}^{[k_i]} & \cdots & \mathbf{J}^{[k_q]T}\boldsymbol{\Omega}^{[k]}\mathbf{J}^{[k_q]} & \\ & & & & & & \ddots \end{pmatrix}$$

$$\mathbf{b}^{[k]T} = \begin{pmatrix} \vdots \\ \mathbf{J}^{[k_1]T}\boldsymbol{\Omega}^{[k]}\mathbf{e}^{[k]} \\ \vdots \\ \mathbf{J}^{[k_i]T}\boldsymbol{\Omega}^{[k]}\mathbf{e}^{[k]} \\ \vdots \\ \mathbf{J}^{[k_q]T}\boldsymbol{\Omega}^{[k]}\mathbf{e}^{[k]} \\ \vdots \end{pmatrix}$$

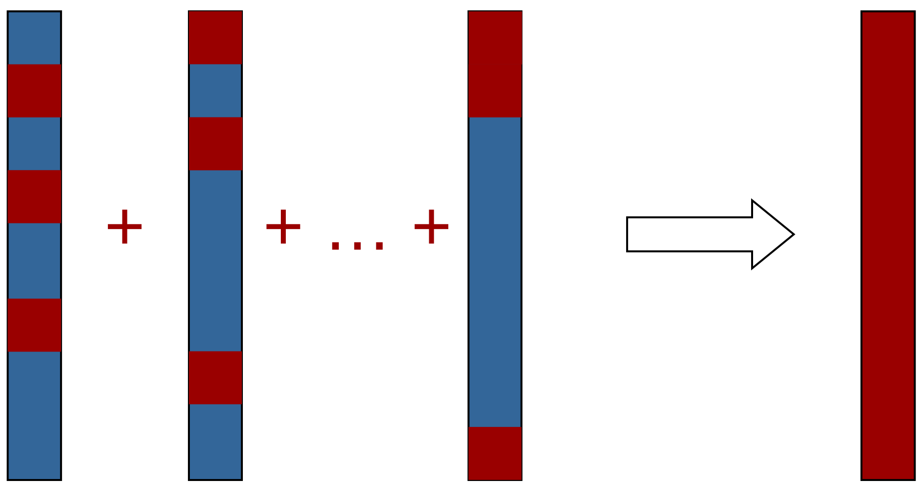
# Structure

Structure of **H**: location of the non zero elements

The structure of H depends only on the structure of the measurements

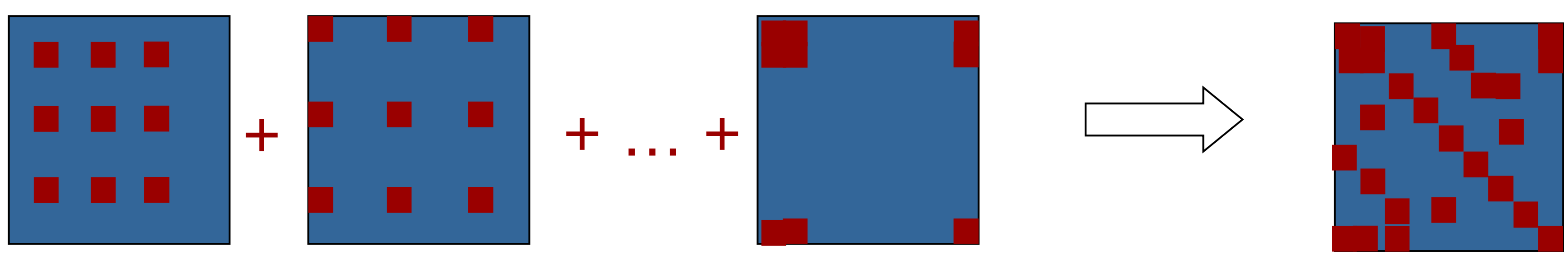
- preallocate before the iterations
- do not allocate memory for zero blocks

$$\mathbf{b} = \sum \mathbf{b}^{[k]}$$



Full diagonal.
each measurement correlates a fixed number of variables with $q^2$ blocks

$$\mathbf{H} = \sum \mathbf{H}^{[k]}$$



# Structure and Efficiency

The number of non-zero blocks in H depends on the number of measurements

Bounded by:

- number of poses
- perception range
- landmark density

In typical SLAM-related problems the number of measurements grows linearly with the length of the trajectory

H has a *linear* number of non-zero blocks (it is mostly empty)

Use special techniques to solve sparse linear systems

# Solving Sparse Systems

A common solution to solve a symmetric positive definite system of equation is through Cholesky factorization

$$\mathbf{H}\boldsymbol{\Delta}\mathbf{x} = -\mathbf{b}$$

System we want to solve

$$\mathbf{H} = \mathbf{L}\mathbf{L}^T$$

Cholesky (L lower triangular)

$$\mathbf{L}\underbrace{\mathbf{L}^T\boldsymbol{\Delta}\mathbf{x}}_{\mathbf{y}} = -\mathbf{b}$$

$$\mathbf{L}\mathbf{y} = -\mathbf{b}$$

Solve for y by forward substitution

$$\mathbf{L}^T\boldsymbol{\Delta}\mathbf{x} = \mathbf{y}$$

Solve for x by backward substitution

**Big Issue**: H sparse does not mean L sparse!!
We lose the benefits of sparsity

# Permutations and Cholesky

Permutation matrix

- Encodes a reordering of the variables
- The elements are either 0 or 1
- Exactly 1 non zero for each row
- Exactly 1 non zero for each column
- The inverse of a permutation is its transpose

If a matrix is sparse, there is a reordering of the variables that renders the Cholesky factor maximally sparse

Computing such an ordering is **NP complete**

Efficient heuristics solve the problem

# Permutations and Cholesky

Solve the linear system through a permutation

$$\mathbf{H}\mathbf{\Delta x} = -\mathbf{b}$$

System we want to solve

$$\mathbf{P}\mathbf{H}\underbrace{\mathbf{P}^T\mathbf{P}}_{\mathbf{I}}\mathbf{\Delta x} = -\mathbf{P}\mathbf{b}$$

Apply a permutation

$$\underbrace{\mathbf{P}\mathbf{H}\mathbf{P}^T}_{\mathbf{H}'}\underbrace{\mathbf{P}\mathbf{\Delta x}}_{\mathbf{\Delta x}'} = -\underbrace{\mathbf{P}\mathbf{b}}_{\mathbf{b}'}$$

$$\mathbf{H}'\mathbf{\Delta x}' = -\mathbf{b}'$$

Solve the system under permutation

$$\mathbf{H}' = \mathbf{L}'\mathbf{L}'^T$$

Cholesky decomposition of H' (sparse)

$$\mathbf{L}'\underbrace{\mathbf{L}'^T\mathbf{\Delta x}'}_{\mathbf{y}} = -\mathbf{b}'$$

$$\mathbf{L}'\mathbf{y} = -\mathbf{b}'$$

solve through forward/backward subst.

$$\mathbf{L}'^T\mathbf{\Delta x}' = \mathbf{y}$$

$$\mathbf{\Delta x} = \mathbf{P}^T\mathbf{\Delta x}'$$

Recover dx applying inverse permutation

# Comments

We approached a complex multi-robot multi-landmark problem

- The measurement independence leads to a sparse structure of H
- The structure of H does not change during the iterations
- It can be efficiently solved by using sparse methods
- Cholesky is not the only possible way (also other approaches such as QR factorization will do)
- Sparse methods rely on finding an ordering that keeps the triangular system sparse

# Total Least Squares

Sparse Optimization with

- Pose-Pose (3D)
- Pose-Landmark (3D)
- Pose-Landmark (Projection: 2D)

Using a factor graph

# Pose-Landmark Constraint

- Here we represent the state as the pose of the robot in the world, i.e. $\mathbf{X} \quad : \quad {}^W\mathbf{T}_R$
- Consequently, prediction and error functions become

$$\mathrm{h}_{\mathrm{icp}}^{[i,j]}(\mathbf{X}) = \mathbf{X}_r^{[i]-1}\mathbf{X}_l^{[j]} = \mathbf{R}_r^{[i]T}(\mathbf{X}_l^{[j]} - \mathbf{t}_r^{[i]})$$

$$\mathrm{e}_{\mathrm{icp}}^{[i,j]}(\mathbf{X}_r^{[i]} \boxplus \mathbf{\Delta x}_r^{[i]}, \mathbf{X}_l^{[j]} + \mathbf{\Delta x}_l^{[j]}) = \mathbf{h}_{\mathrm{icp}}^{[i,j]}(\mathbf{X}_r^{[i]} \boxplus \mathbf{\Delta x}_r^{[i]}, \mathbf{X}_l^{[j]} + \mathbf{\Delta x}_l^{[j]}) - \mathbf{z}_{\mathrm{icp}}^{[i,j]}$$

# Pose-proj Constraint

- Here we represent the state as the pose of the robot in the world, i.e. $\mathbf{X} \quad : \quad {}^{W}\mathbf{T}_{R}$
- Consequently, prediction and error functions become

$$\mathrm{h}_{\mathrm{prj}}^{[i,j]}(\mathbf{X}) = \pi(\mathbf{K}(\mathbf{X}_r^{[i]-1}\mathbf{X}_l^{[j]})) = \pi\left(\mathbf{K}\mathbf{R}_r^{[i]T}(\mathbf{X}_l^{[j]} - \mathbf{t}_r^{[i]})\right)$$

$$\mathrm{e}_{\mathrm{prj}}^{[i,j]}(\mathbf{X}_r^{[i]} \boxplus \mathbf{\Delta x}_r^{[i]}, \mathbf{X}_l^{[j]} + \mathbf{\Delta x}_l^{[j]}) = \mathbf{h}_{\mathrm{prj}}^{[i,j]}(\mathbf{X}_r^{[i]} \boxplus \mathbf{\Delta x}_r^{[i]}, \mathbf{X}_l^{[j]} + \mathbf{\Delta x}_l^{[j]}) - \mathbf{z}_{\mathrm{prj}}^{[i,j]}$$

# Pose-Pose Constraint

- Here we represent the state as the pose of the robot in the world, i.e. $\mathbf{X} \quad : \quad {}^{W}\mathbf{T}_{R}$

- Consequently, prediction and error functions become

$$\mathrm{h}_{\mathrm{p2p}}^{[i,j]}(\mathbf{X}) = \mathrm{flatten}(\mathbf{X}_r^{[i]-1}\mathbf{X}_r^{[j]})$$

$$\mathrm{e}_{\mathrm{p2p}}^{[i,j]}(\mathbf{X}) = \mathbf{h}_{\mathrm{p2p}}^{[i,j]}(\mathbf{X}) - \mathrm{flatten}(\mathbf{Z}_{\mathrm{p2p}}^{[i,j]})$$

$$\mathrm{e}_{\mathrm{p2p}}^{[i,j]}(\mathbf{X}_r^{[i]} \boxplus \mathbf{\Delta x}_r^{[i]}, \mathbf{X}_l^{[j]} \boxplus \mathbf{\Delta x}_l^{[j]}) = \mathbf{h}_{\mathrm{p2p}}^{[i,j]}(\mathbf{X}_r^{[i]} \boxplus \mathbf{\Delta x}_r^{[i]}, \mathbf{X}_l^{[j]} \boxplus \mathbf{\Delta x}_l^{[j]}) - \mathrm{flatten}(\mathbf{Z}_{\mathrm{p2p}}^{[i,j]})$$