Cross interpolation

Definition

For a given matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$, the cross interpolation is defined as:

$$ \mathbf{A} \approx \mathbf{\tilde{A}} = \mathbf{A}(\mathbb{I}, J) \mathbf{A}(I, J)^{-1} \mathbf{A}(I, \mathbb{J}), $$

where $\mathbb{I} = {1, 2, ..., m}$ and $\mathbb{J} = {1, 2, ..., n}$, while $I \subseteq \mathbb{I}$ and $J \subseteq \mathbb{J}$ with $|I| = |J| = r$.

Main properties

- **Interpolation**

$$ \begin{aligned} \mathbf{A}(\mathbb{I}, J) &= \mathbf{\tilde{A}}(\mathbb{I}, J), \ \mathbf{A}(I, \mathbb{J}) &= \mathbf{\tilde{A}}(I, \mathbb{J}). \ \end{aligned} $$

- **Low-rank decomposition**

$$ \mathbf{A} = \mathbf{\tilde{A}} \quad \text{if} \quad \text{rank}(\mathbf{A}) \leq r. $$

Algorithm

First, define an error function (matrix) as:

$$ \begin{aligned} \mathcal{E}(i, j) &= \text{abs}(\mathbf{A} - \mathbf{\tilde{A}})(i, j) \ &= \text{abs}(\mathbf{A}(i, j) - \mathbf{\tilde{A}}(i, j)) \ &= \text{abs}(\mathbf{A}(i, j) - \mathbf{A}(i, J) \mathbf{A}(I, J)^{-1} \mathbf{A}(I, j)) \ \end{aligned} $$

where $i \in \mathbb{I} \setminus I$ and $j \in \mathbb{J} \setminus J$.

Full Search

Here, it first implements a `full search` algorithm to find the optimal sets I and J.

Then the algorithm goes as follows:

1. Find a point (i, j) that maximizes $\mathcal{E}(i, j)$:

$$ (i^, *j*^) = \argmax_{i \in \mathbb{I}, j \in \mathbb{J}} \mathcal{E}(i, j). $$

2. Add the point $(i^, *j*^)$ to the sets I and J:

$$ I \leftarrow I \cup {i^*}, \quad J \leftarrow J \cup {j*^}. $$

3. Repeat steps 1 and 2 until the interpolation error satisfies a predefined threshold:

$$ |\mathbf{\mathcal{E}}|_{\xi} < \epsilon, $$

where $|\cdot|_{\xi}$ is a norm (e.g., Frobenius norm) and ϵ is a predefined threshold.

Rook Search

Then, implements the `rook search` algorithm goes as:

1. Randomly initialize a new pivot $(i^, j^)$.
2. Column-wise movement:

$$ i^* \leftarrow \argmax_{i \in \mathbb{I}} \mathcal{E}(i, j^*). $$

3. Row-wise movement:

$$ j^* \leftarrow \argmax_{j \in \mathbb{J}} \mathcal{E}(i^*, j). $$

4. Repeat steps 2 and 3 until the limit of iterations is reached or the following condition (rook condition) is met:

$$ \begin{aligned} i^* &= \argmax_{i \in \mathbb{I}} \mathcal{E}(i, j^), \ j^ &= \argmax_{j \in \mathbb{J}} \mathcal{E}(i^*, j). \end{aligned} $$

5. Add the point $(i^, j^)$ to the sets I and J:

$$ I \leftarrow I \cup {i^}, \quad J \leftarrow J \cup {j^}. $$

6. Repeat steps 1 to 5 until the interpolation error satisfies a predefined threshold:

$$ |\mathbf{\mathcal{E}}|_{\xi} < \epsilon. $$

Implementation tips

Schur Complement

Given a invertible matrix $\mathbf{U} \in \mathbb{R}^{k \times k}$ and the inverse $\mathbf{U}^{-1}$, and consider the following block matrix

$$ \mathbf{M} = \begin{bmatrix} \mathbf{U} & \mathbf{c} \ \mathbf{r} & p \end{bmatrix} \in \mathbb{R}^{(k + 1) \times (k + 1)} $$

where

$$ \begin{aligned} \mathbf{c} &\in \mathbb{R}^{k \times 1}, \ \mathbf{r} &\in \mathbb{R}^{1 \times k}, \ p &\in \mathbb{R}. \end{aligned} $$

Then the inverse of $\mathbf{M}$ can be computed as

$$ \begin{aligned} \mathbf{M}^{-1} &= \begin{bmatrix} \mathbf{U}^{-1} + \mathbf{U}^{-1} \mathbf{c} s^{-1} \mathbf{r} \mathbf{U}^{-1} & -\mathbf{U}^{-1} \mathbf{c} s^{-1} \ -s^{-1} \mathbf{r} \mathbf{U}^{-1} & s^{-1} \end{bmatrix} \in \mathbb{R}^{(k + 1) \times (k + 1)} \ &= \begin{bmatrix} \mathbf{U}^{-1} + \mathbf{l} s^{-1} \mathbf{h} & -\mathbf{l} s^{-1} \ -s^{-1} \mathbf{h} & s^{-1} \end{bmatrix} \end{aligned} $$

where

$$ \begin{aligned} \mathbf{l} &= \mathbf{U}^{-1} \mathbf{c} \in \mathbb{R}^{k \times 1}, \ \mathbf{h} &= \mathbf{r} \mathbf{U}^{-1} \in \mathbb{R}^{1 \times k}, \ s &= p - \mathbf{r} \mathbf{U}^{-1} \mathbf{c} \in

\mathbb{R}. \end{aligned} $$

- In the above equations, s is actually the Schur complement of $\mathbf{U}$ in $\mathbf{M}$, and the matrix $\mathbf{M}$ is singular when $s = 0$.
- The above equations only hold when $\mathbf{U}$ and $\mathbf{M}$ are both invertible.

ACA (Adaptive Cross Approximation)

At step $(t-1)$, one have

$$ \begin{aligned} \mathbf{C}{t - 1} &= \mathbf{A}(\mathbb{I}, J{t-1}), \ \mathbf{U}{t - 1}^{-1} &= \mathbf{A}(I{t-1}, J_{t-1})^{-1}, \ \mathbf{R}{t - 1} &= \mathbf{A}(I{t-1}, \mathbb{J}), \ \mathbf{\tilde{A}}{t - 1} &= \mathbf{C}{t - 1} \mathbf{U}{t - 1}^{-1} \mathbf{R}{t - 1}. \end{aligned} $$

Now, one somehow acquire a new pivot at $(i^, j^)$, so that one have

$$ \begin{aligned} \mathbf{C}{t} &= \begin{bmatrix} \mathbf{C}{t-1} & \mathbf{A}(:, j^*) \end{bmatrix} \in \mathbb{R}^{m \times t}, \

```
\mathbf{U}_{t} &= \begin{bmatrix}
    \mathbf{U}_{t - 1} & \mathbf{c} \\
    \mathbf{r} & p
\end{bmatrix} \in \mathbb{R}^{t \times t}, \\

\mathbf{c} &= \mathbf{A}(I_{t - 1}, j^*) \in \mathbb{R}^{(t - 1) \times 1}, \\

\mathbf{r} &= \mathbf{A}(i^*, J_{t - 1}) \in \mathbb{R}^{1 \times (t - 1)}, \\

p &= \mathbf{A}(i^*, j^*) \in \mathbb{R}, \\

\mathbf{R}_{t} &= \begin{bmatrix}
    \mathbf{R}_{t - 1} \\
    \mathbf{A}(i^*, :)
\end{bmatrix} \in \mathbb{R}^{t \times n}, \\
```

\end{aligned} $$

and then

$$ \begin{aligned} \mathbf{\tilde{A}}{t} &= \mathbf{C}{t} \mathbf{U}{t}^{-1} \mathbf{R}{t} \ &= \begin{bmatrix} \mathbf{C}{t-1} & \mathbf{A}(:, j^*) \end{bmatrix} \begin{bmatrix} \mathbf{U}{t - 1}^{-1} + \mathbf{U}{t - 1}^{-1} \mathbf{c} s^{-1} \mathbf{r} \mathbf{U}{t - 1}^{-1} & -\mathbf{U}{t - 1}^{-1} \mathbf{c} s^{-1} \ -s^{-1} \mathbf{r} \mathbf{U}{t - 1}^{-1} & s^{-1} \end{bmatrix} \begin{bmatrix} \mathbf{R}{t - 1} \ \mathbf{A}(i^*, 😃 \end{bmatrix} \ &= \mathbf{C}{t - 1} \mathbf{U}{t - 1}^{-1} \mathbf{R}{t - 1} + s^{-1} (\mathbf{C}{t - 1} \mathbf{U}{t - 1}^{-1} \mathbf{c} \mathbf{r} \mathbf{U}{t - 1}^{-1} \mathbf{R}{t - 1} - \mathbf{A}(:, j^) \mathbf{r} \mathbf{U}{t - 1}^{-1} \mathbf{R}{t - 1} - \mathbf{C}{t - 1} \mathbf{U}{t - 1}^{-1} \mathbf{c} \mathbf{A}(i^, 😃 + \mathbf{A}(:, j^) \mathbf{A}(i^, 😃) \ &= \mathbf{\tilde{A}}{t - 1} + s^{-1} (\mathbf{\tilde{A}}{t - 1}(:, j^) \mathbf{\tilde{A}}_{t - 1}(i^, 😃 - \mathbf{A}(:, j^) \mathbf{\tilde{A}}_{t - 1}(i^, 😃

- \mathbf{\tilde{A}}{t - 1}(:, j^) \mathbf{A}(i^, 😃 + \mathbf{A}(:, j^) \mathbf{A}(i^, 😃) \ &= \mathbf{\tilde{A}}{t - 1} + s^{-1} (\mathbf{A}(:, j^) - \mathbf{\tilde{A}}_{t - 1}(:, j^)) (\mathbf{A}(i^, 😃 - \mathbf{\tilde{A}}_{t - 1}(i^, 😃). \end{aligned} $$

Note that

$$ \begin{aligned} s = p - \mathbf{r} \mathbf{U}{t - 1}^{-1} \mathbf{c} = \mathbf{A}(i^, j^) - \mathbf{\tilde{A}}{t - 1}(i^, j^) = \mathcal{E}{t - 1}(i^, j^), \ \mathbf{A}(:, j^*) - \mathbf{\tilde{A}}{t - 1}(:, j^) = \mathcal{E}_{t - 1}(:, j^), \ \mathbf{A}(i^, 😃 - \mathbf{\tilde{A}}_{t - 1}(i^, 😃 = \mathcal{E}_{t - 1}(i^*, 😃, \end{aligned} $$

therefore

$$ \mathbf{\tilde{A}}{t} = \mathbf{\tilde{A}}{t - 1} + \frac{1}{\mathcal{E}{t - 1}(i^, j^)} \mathcal{E}{t - 1}(:, j^) \mathcal{E}_{t - 1}(i^, 😃. $$

With the above equation, one can establish the following algorithm:

1. Initialize $\mathbf{\tilde{A}}{0} = \mathbf{0}, \mathcal{E}{0} = \mathbf{A}$.
2. Pick a new pivot $(i^, j^)$.
3. Update the matrices:
   - $\mathcal{D}{t - 1} = \frac{1}{\mathcal{E}{t - 1}(i^, j^)} \mathcal{E}{t - 1}(:, j^*) \mathcal{E}{t - 1}(i^*, :)$;
   - $\mathbf{\tilde{A}}{t} = \mathbf{\tilde{A}}{t - 1} + \mathcal{D}_{t - 1}$;
   - $\mathcal{E}{t} = \mathcal{E}{t - 1} - \mathcal{D}_{t - 1}$.

Efficient ACA

In some scenarios, one should assume that the original matrix $\mathbf{A}$ is too large to fit in memory and expensive to evaluate. In such cases, one should follows a more memory-efficient implementation.

Let

$$ \begin{aligned} p_t &= \mathcal{E}_{t}(I_k, J_k), \ \mathbf{c}t &= \mathcal{E}{t}(:, J_k), \ \mathbf{r}t &= \mathcal{E}{t}(I_k, 😃, \end{aligned} $$

then one has

$$ \begin{aligned} \mathcal{E}{t} &= \mathbf{A} - \mathbf{\tilde{A}}{t} = \mathbf{A} - \sum_{k = 0}^{t - 1} \frac{1}{p_{k}} \mathbf{c}{k} \mathbf{r}{k}, \ \mathcal{E}{t}(:, j) &= \mathbf{A}(:, j) - \mathbf{\tilde{A}}{t}(:, j) = \mathbf{A}(:, j) - \sum_{k = 0}^{t - 1} \frac{1}{p_{k}} \mathbf{c}{k} \mathbf{r}{k}(j), \ \mathcal{E}{t}(i, 😃 &= \mathbf{A}(i, 😃 - \mathbf{\tilde{A}}{t}(i, 😃 = \mathbf{A}(i, 😃 - \sum_{k = 0}^{t - 1} \frac{1}{p_{k}} \mathbf{c}{k}(i) \mathbf{r}{k}. \end{aligned} $$

Therefore, one can follows

1. Find a new pivot $(i^, j^)$.
2. Find new vectors:
   - $\mathbf{c}{t} = \mathcal{E}{t}(:, j^*)$;
   - $\mathbf{r}{t} = \mathcal{E}{t}(i^*, :)$.