拡散方程式における並列用反復解法

2024年6月13日

後　保範（ISCPC)

1. はじめに

- 拡散方程式におけるCG法の前処理
- 加速パラメータ（ω）をICに追加可能?

  D = ω/(対角 - IC処理)　： IC処理は正

  　D×対角 > 2なら収束しない

  　D×対角 = ω < 2　： SSOR
- 上限パラメータ（γ）の導入が必要

  　D×対角 <= γに保つ、γ=1.94など2未満

注）IC: 不完全コレスキー分解、CG法:共役勾配法

SSOR: 対称逐次加速緩和法

1.1 はじめに(追加)

- 予稿集での収束の速さ評価
  SICはω=1.48、γ=1.94の固定パラメータ
  MIC、SSOR、SICを測定評価
- 発表での収束の速さ評価
  予稿集の評価に加え
  SICのω、γはプログラムで自動設定
  SSOR、SIC(固定)、SIC(自動)を測定評価
  注) SIC: 提案解法

2. 前処理付き共役勾配法(CG)

- 共役勾配法(CG)は対称疎行列用の反復解法
- M行列(拡散方程式)は前処理で収束が良くなる
- 前処理はSSOR、IC(不完全コレスキー)が有名
- MIC(改定IC)はICにパラメータ(α)を追加
- スカラー計算ではMICが最も収束が速い
- 並列計算用の前処理法を提案

加速(ω)と上限(γ)をICに付けた前処理法

SIC(SSOR的IC)と名付ける。

2.1 前処理付きCG法の手順

初期値$x=0$,　行列Aを不完全LDL^T分解する

$r=b-Ax$,　$p=(LDL^T)^{-1}r$,　$\mu_1=(r,p)$

前処理手法が決まる

収束するまで以下を反復計算

$q=Ap$,　$\alpha=\mu_1/(p,q)$

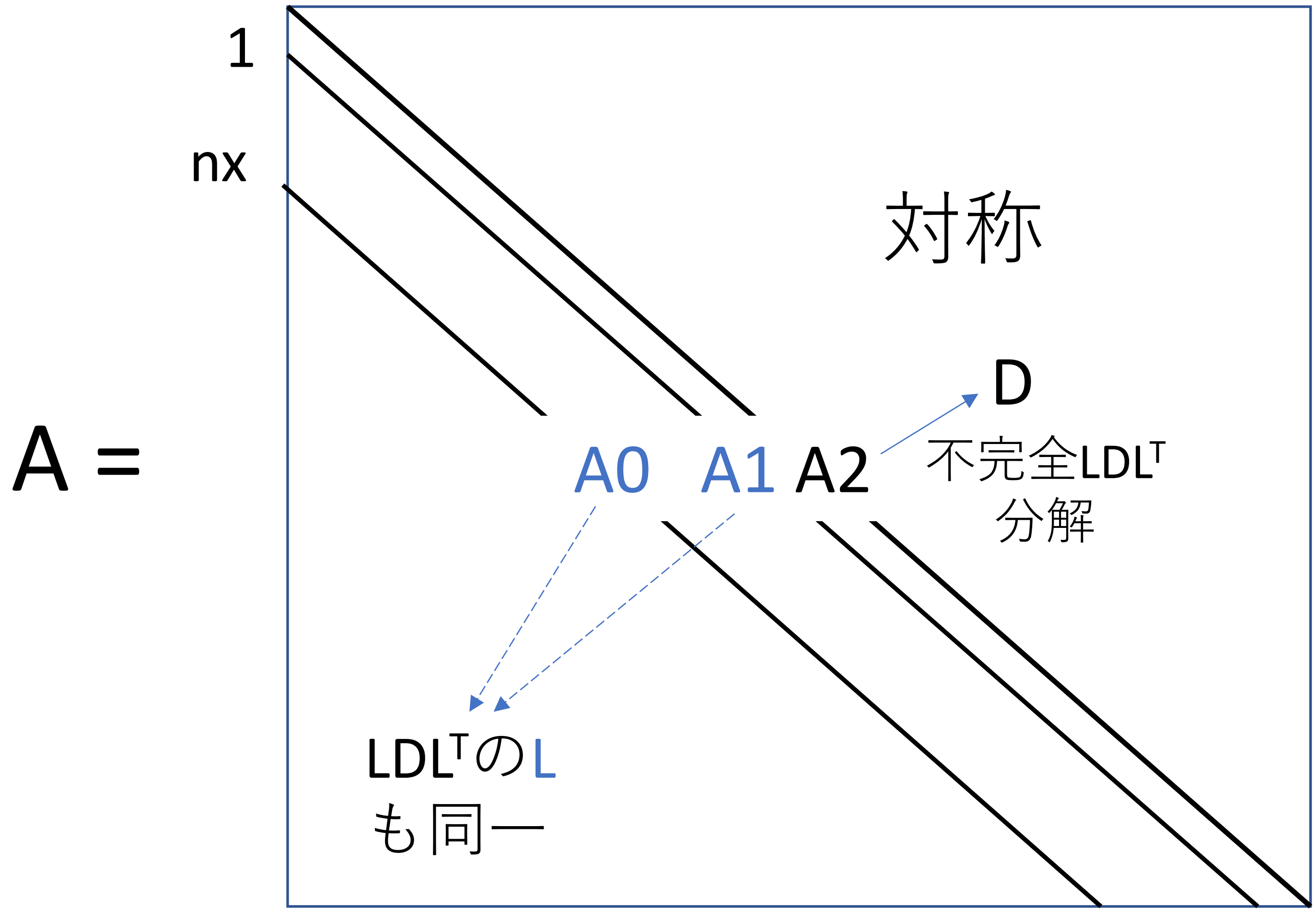
$x=x+\alpha p$,　$r=r-\alpha q$

$q=(LDL^T)^{-1}r$,　$\mu_2=(r,q)$

$\beta=\mu_2/\mu_1$,　$\mu_1=\mu_2$

$p=q+\beta p$

2.3 ２次元差分法による行列



2.4 $q=(LDL^T)^{-1}r$の計算(共通)

```
for (k=0; k<n; k++)
   { q[k] = D[k]*( r[k] - A1[k]*q[k-1]
                          - A0[k]*q[k-nx] ) ;
   }
for (k=n-1; k>=0; k--)
   { q[k] -= D[k]*( A1[k+1]*q[k+1]
                   + A0[k+nx]*q[k+nx] ) ;
   }
```

2.5 不完全LDL^{T}分解(SSOR)

- SSORの加速係数ω(<2.0)、Dを求める。

```
for (k=0; k<n; k++)
    { D[k] = ω / A2[k]; }
```

注)

不完全LDL^{T}分解では対角を除くLは

A0,A1をそのまま使用(IC,MICも)

2.6 不完全LDL^{T}分解(IC)

- パラメータ無し、Dを求める。分解誤差無視。

```
for (k=0; k<n; k++)
  { dv = A2[k] - A1[k]*A1[k]*D[k-1]
               - A0[k]*A0[k]*D[k-nx];
    D[k] = 1.0 / dv;
  }
```

2.7 不完全LDL^{T}分解(MIC)

- パラメータα、Dを求める。対角−α*誤差

```
for (k=0; k<n; k++)
  { dv = A2[k] - A1[k]*D[k-1]*( A1[k] +
                                    α*A0[k+nx-1] ) ;
                - A0[k]*D[k-nx]*( A0[k] +
                                    α*A1[k-nx+1] ) ;
    D[k] = 1.0 / dv;
  }
```

2.8 不完全LDL^{T}分解(SIC)

- パラメータωとγ、Dを求める。

```
for (k=0; k<n; k++)
  { dv = A2[k] - A1[k]*A1[k]*D[k-1]
               - A0[k]*A0[k]*D[k-nx];
    hi = ω*A2[k] / dv;         --- 要素加速
    if (hi > γ) hi = γ;        --- 上限
    D[k] = hi / A2[k];         --- ω/dv
  }
```

3 計算対象(環境)

- 計算機

  Intel Core i7 6700k, 4Ghz, 8GBメモリ
- OS: Windows10
- コンパイラ

  Cygwin Ver. 2.7 gcc, -O3
- 拡散方程式(-div(k・grad(φ))=f)の差分法

  離散化(2次元) --- 並列単位に配列作成
- 収束判定(下記となる反復回数)

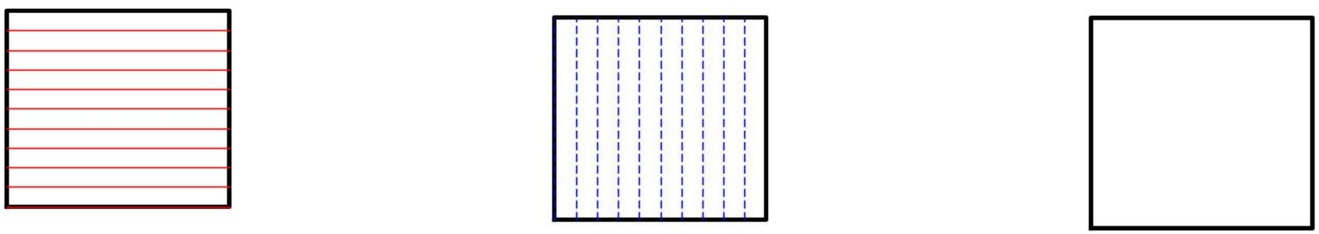
  $||Ax-b||_2 \,/\, ||b||_2 <= 1/10^6$

3.1 2次元モデルの係数

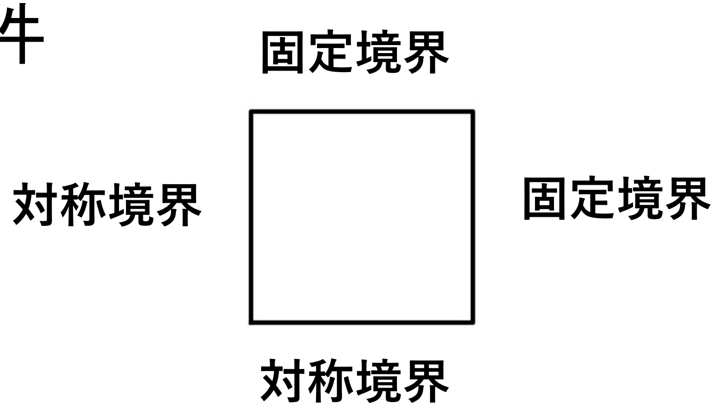
- 拡散方程式(-div(grad(k・ϕ))の係数

k = 0.5　　k = 0.1　　k = 1.0

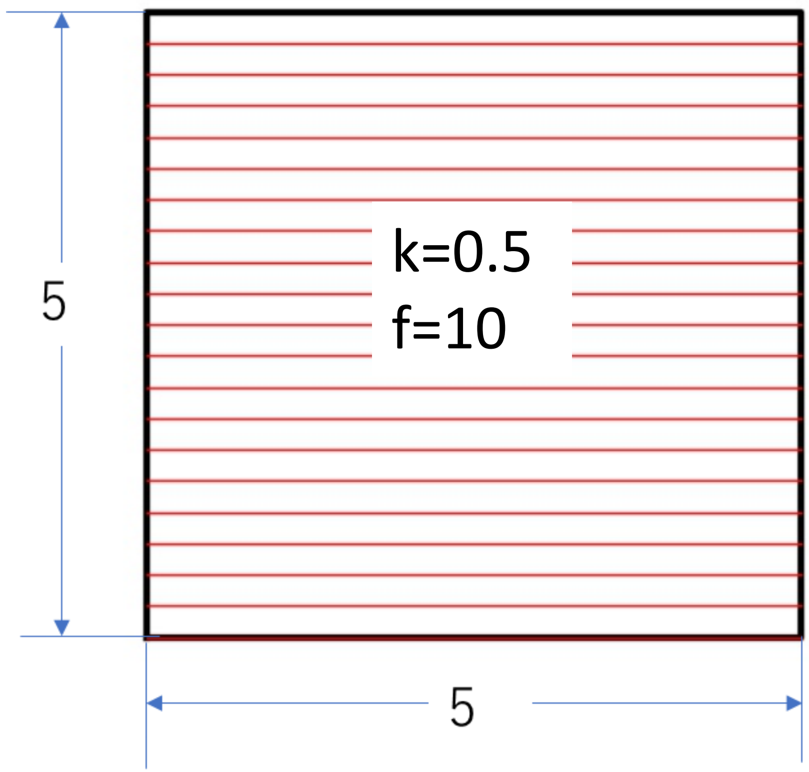
f = 10　　f = 0　　f = 0



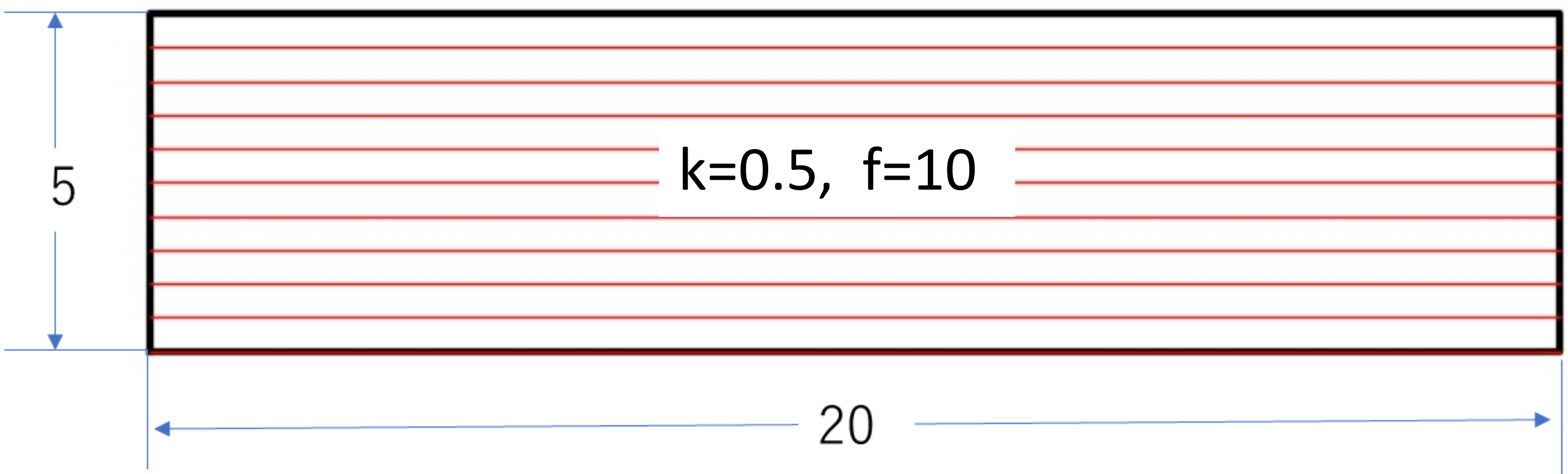
- 境界条件



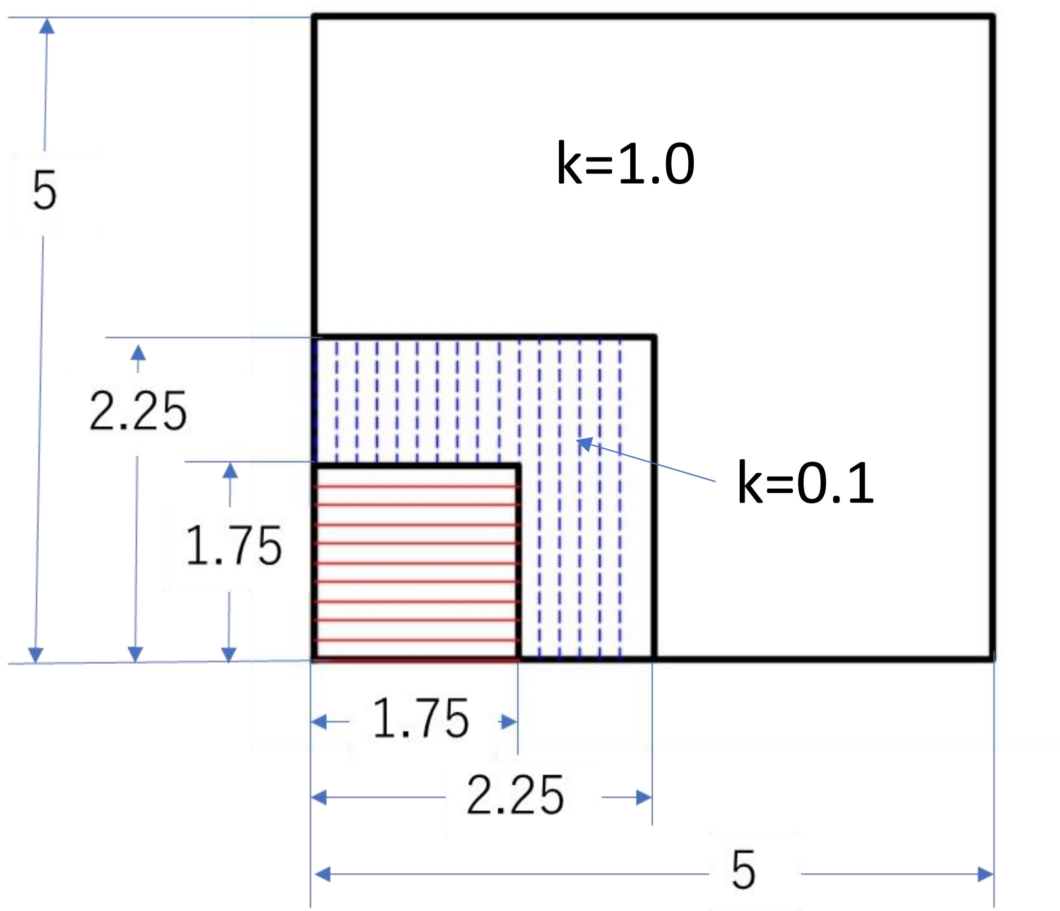
3.2 2次元モデル1(M2A)



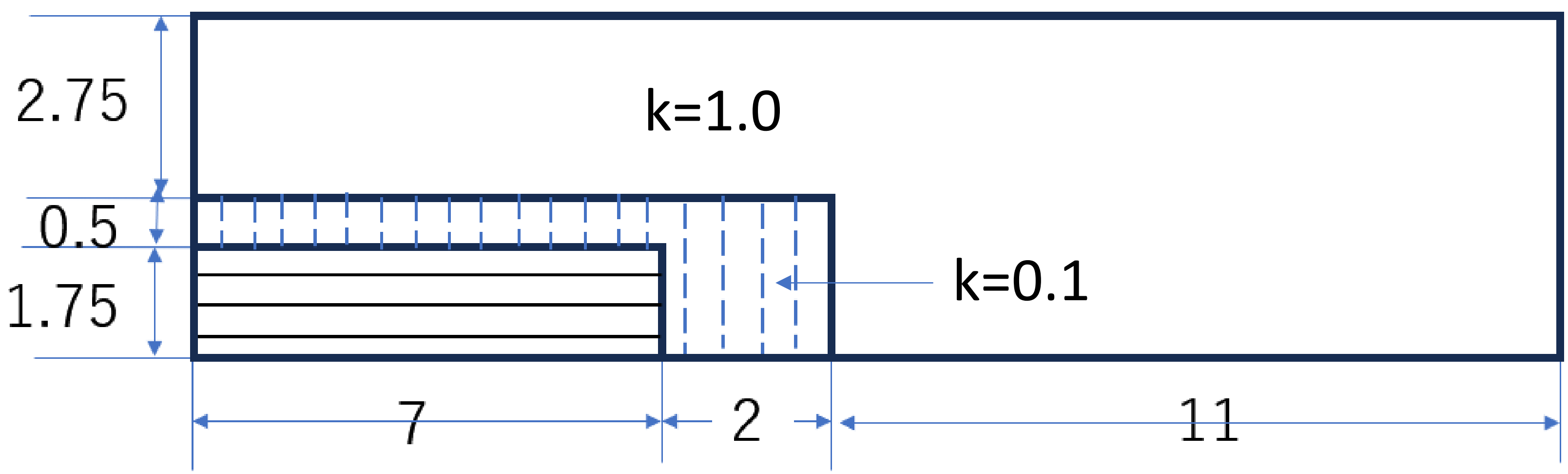
3.3 2次元モデル2(M2B)



3.4 2次元モデル3(M2C)



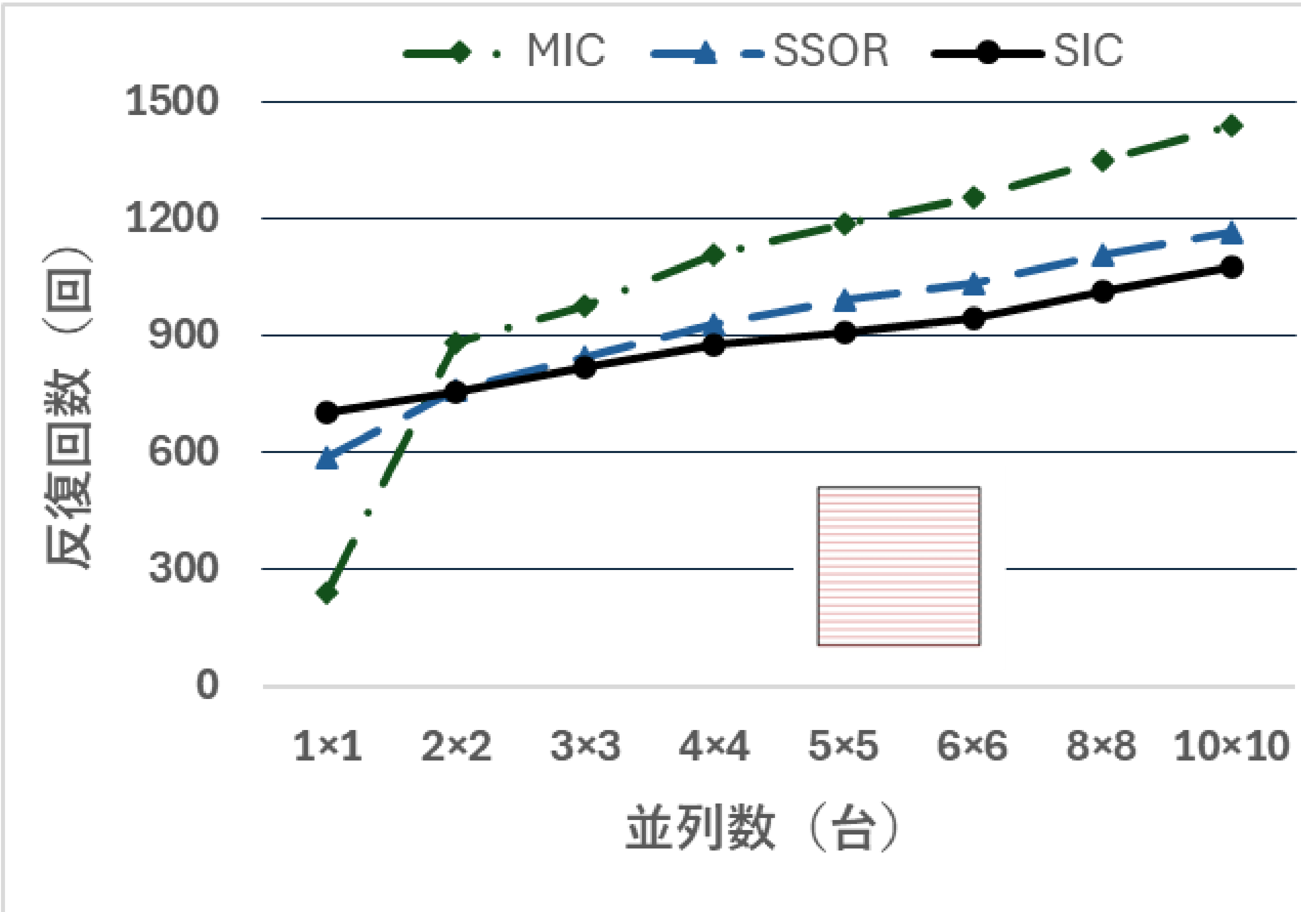
3.5 2次元モデル4(M2D)



4 SSOR, MIC, SIC付きCG法の比較

- 拡散方程式に差分法の反復法($Ax=b$)で比較
- 比較は$||Ax-b||_2/||b||_2<10^{-6}$となる反復数
- SSOR, MICの前処理付きCG法と比較
- SSORは加速係数を$\omega=\omega+0.01$で変化させ、反復数が最も少ないものを使用
- MICはパラメータαを$1-\alpha=(1-\alpha)/2$で変化させ、反復数が最も少ないものを使用
- SICはパラメタω(加速)とγ(上限値)を固定
  $\omega=1.48$,　$\gamma=1.94$

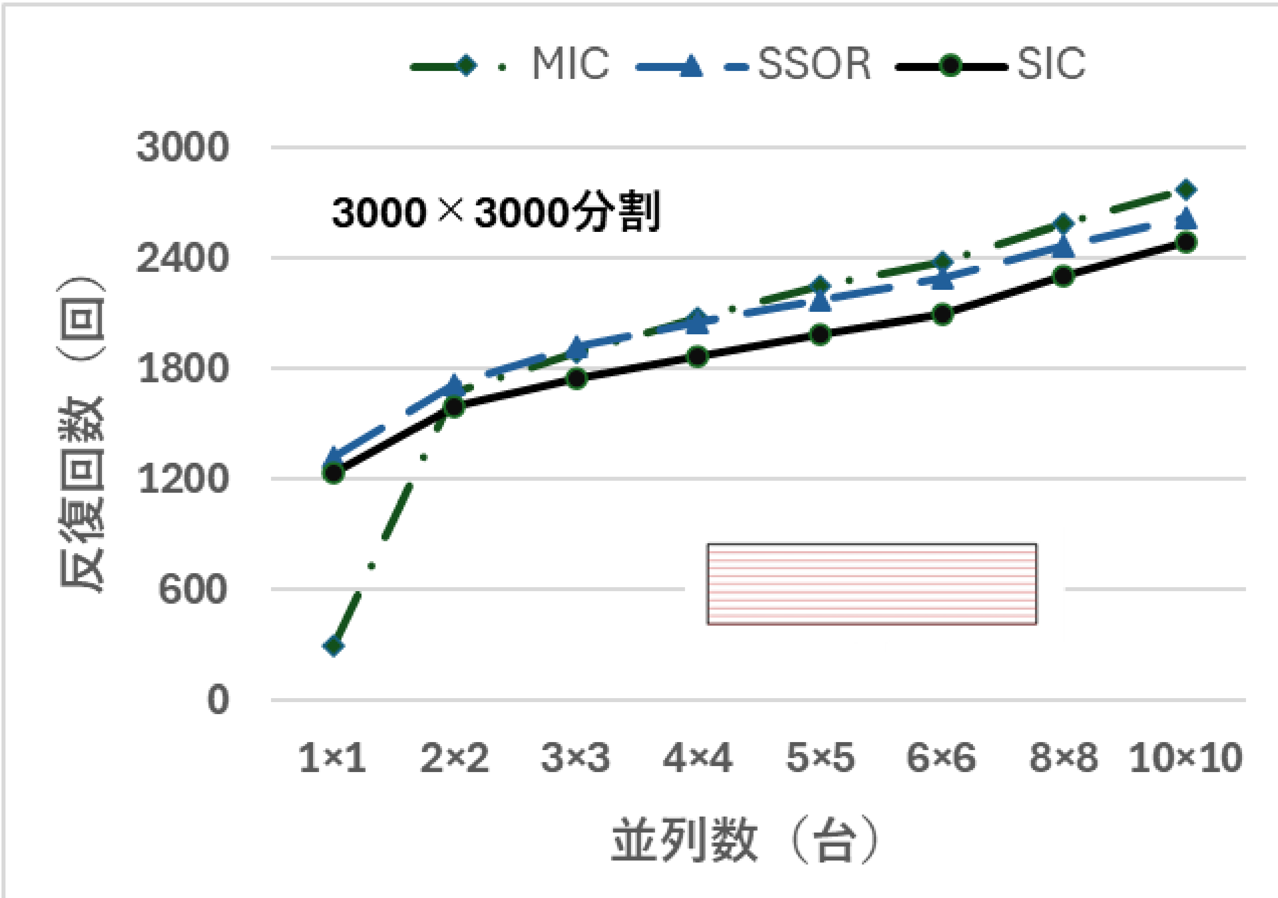
4.1 収束比較(並列数、M2A)



3000×3000分割

• ω=1.48, γ=1.94

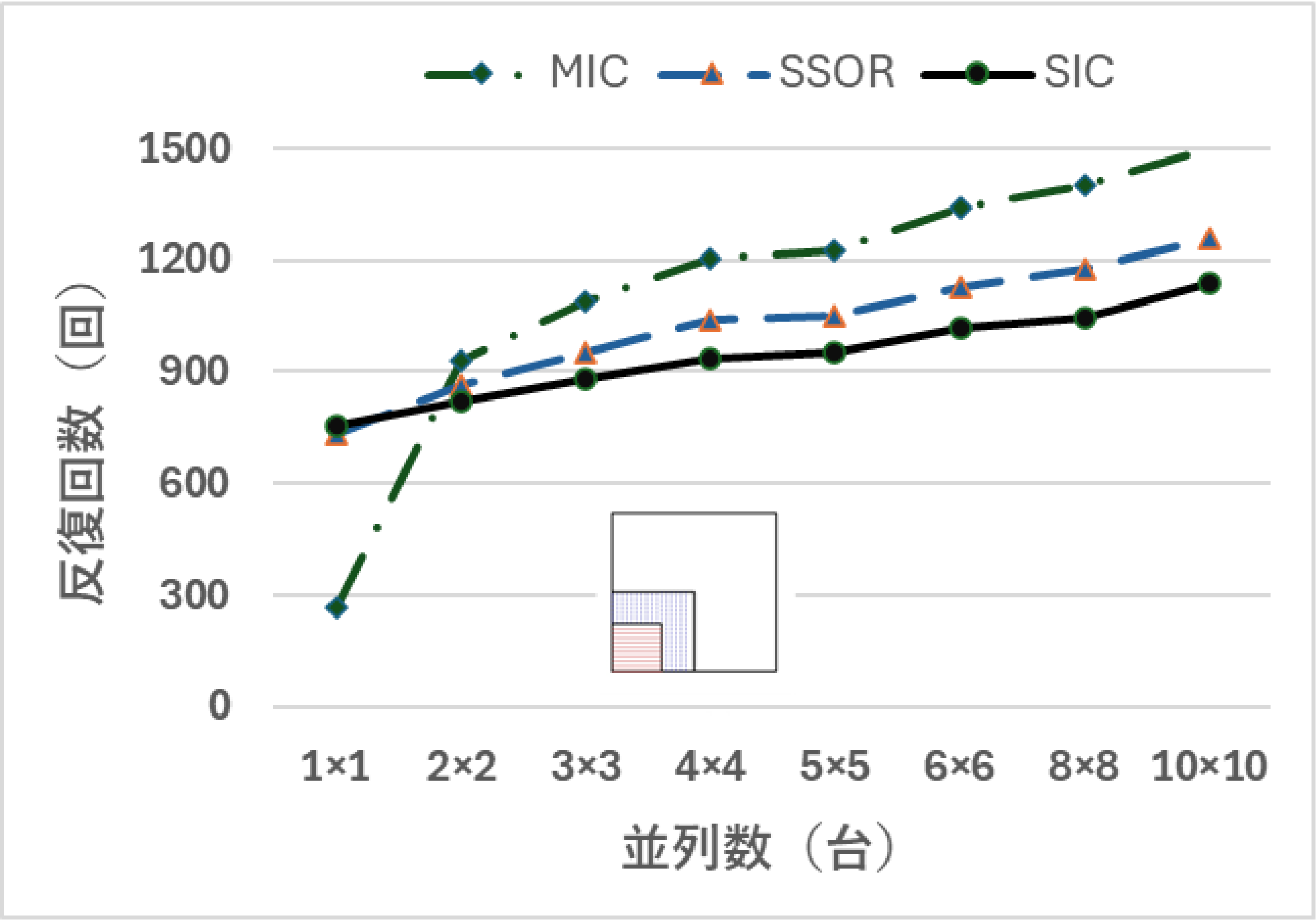
4.2 収束比較(並列数、M2B)



3000×3000分割

- ω=1.48,　γ=1.94

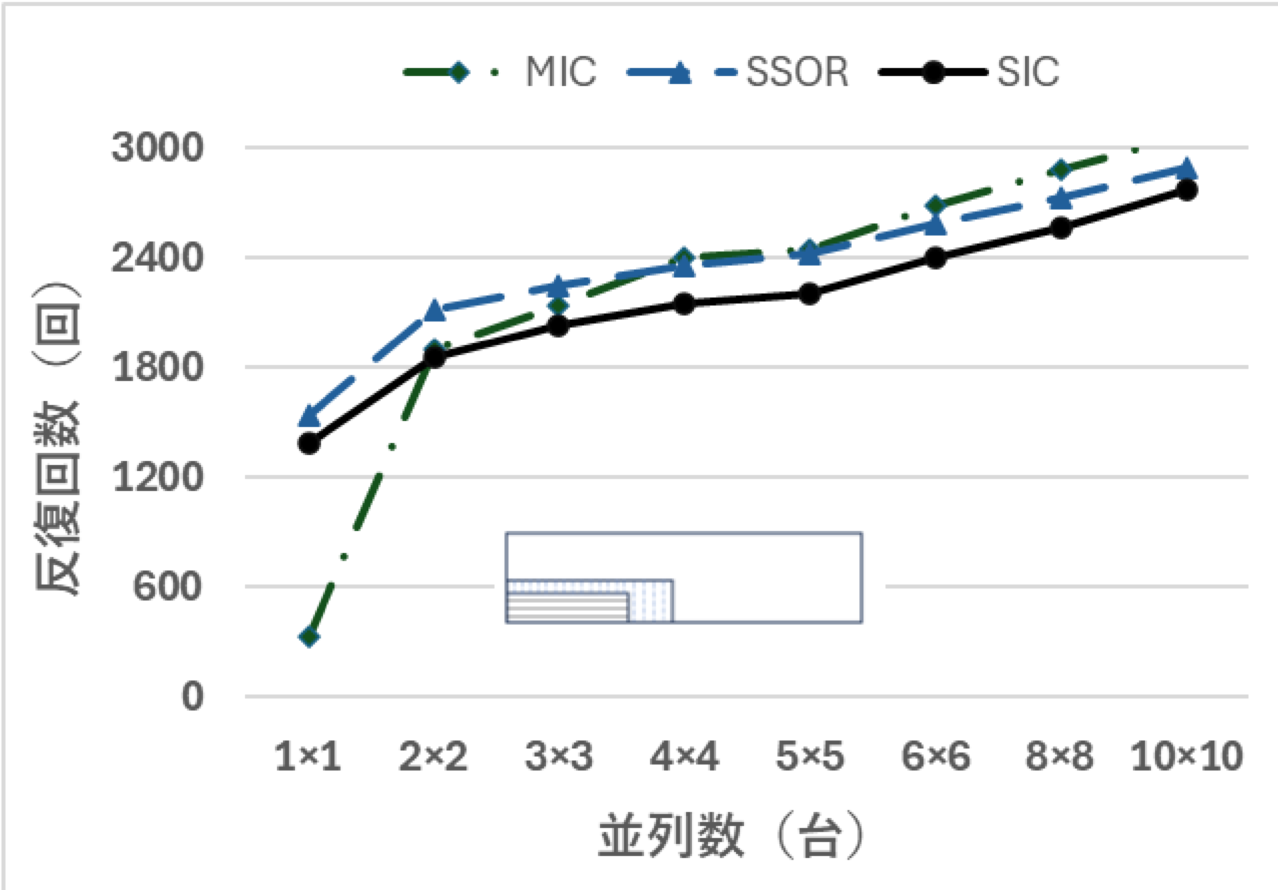
4.3 収束比較(並列数、M2C)



3000×3000分割

- ω=1.48, γ=1.94

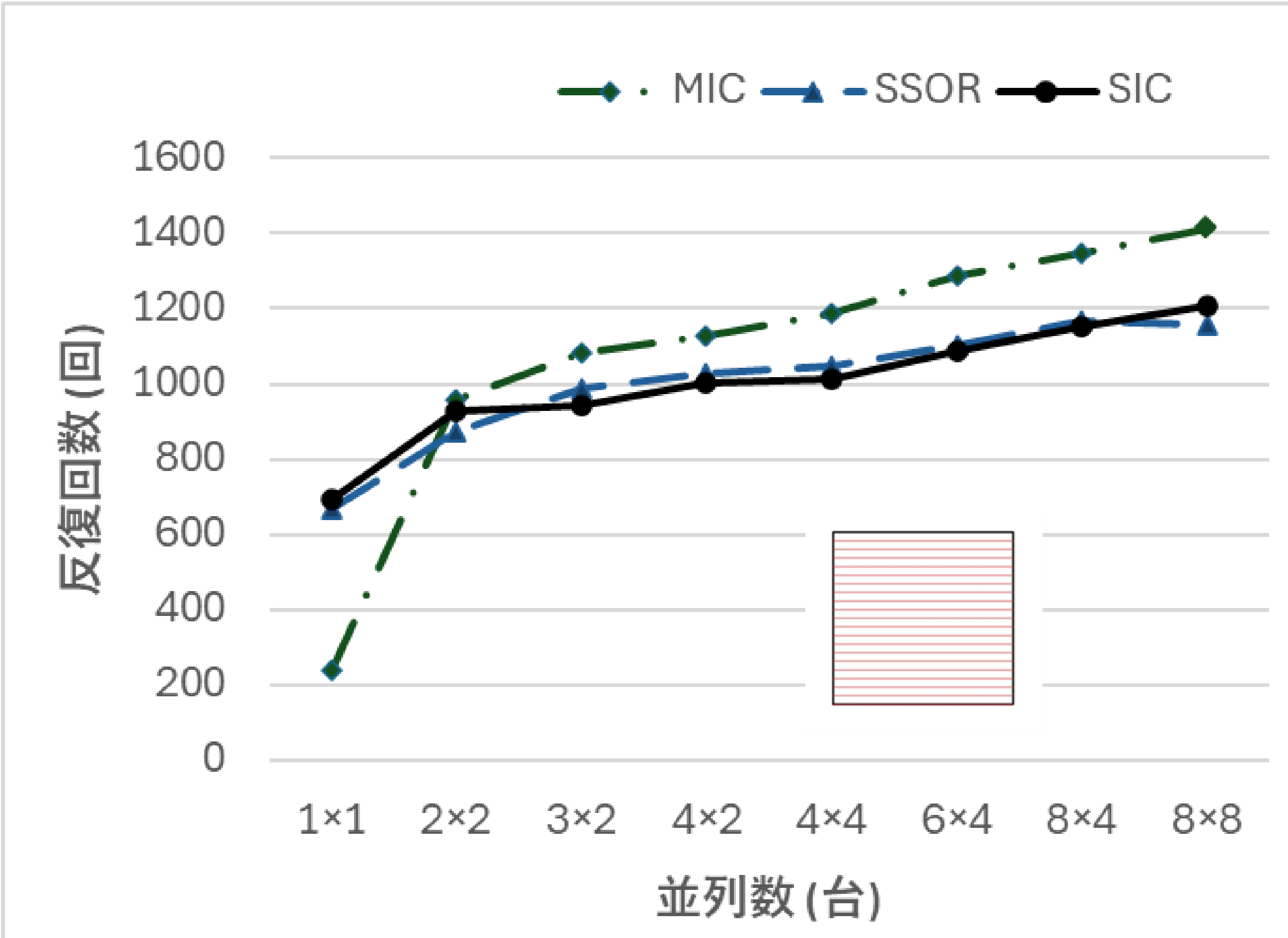
4.4 収束比較(並列数、M2D)



3000×3000分割

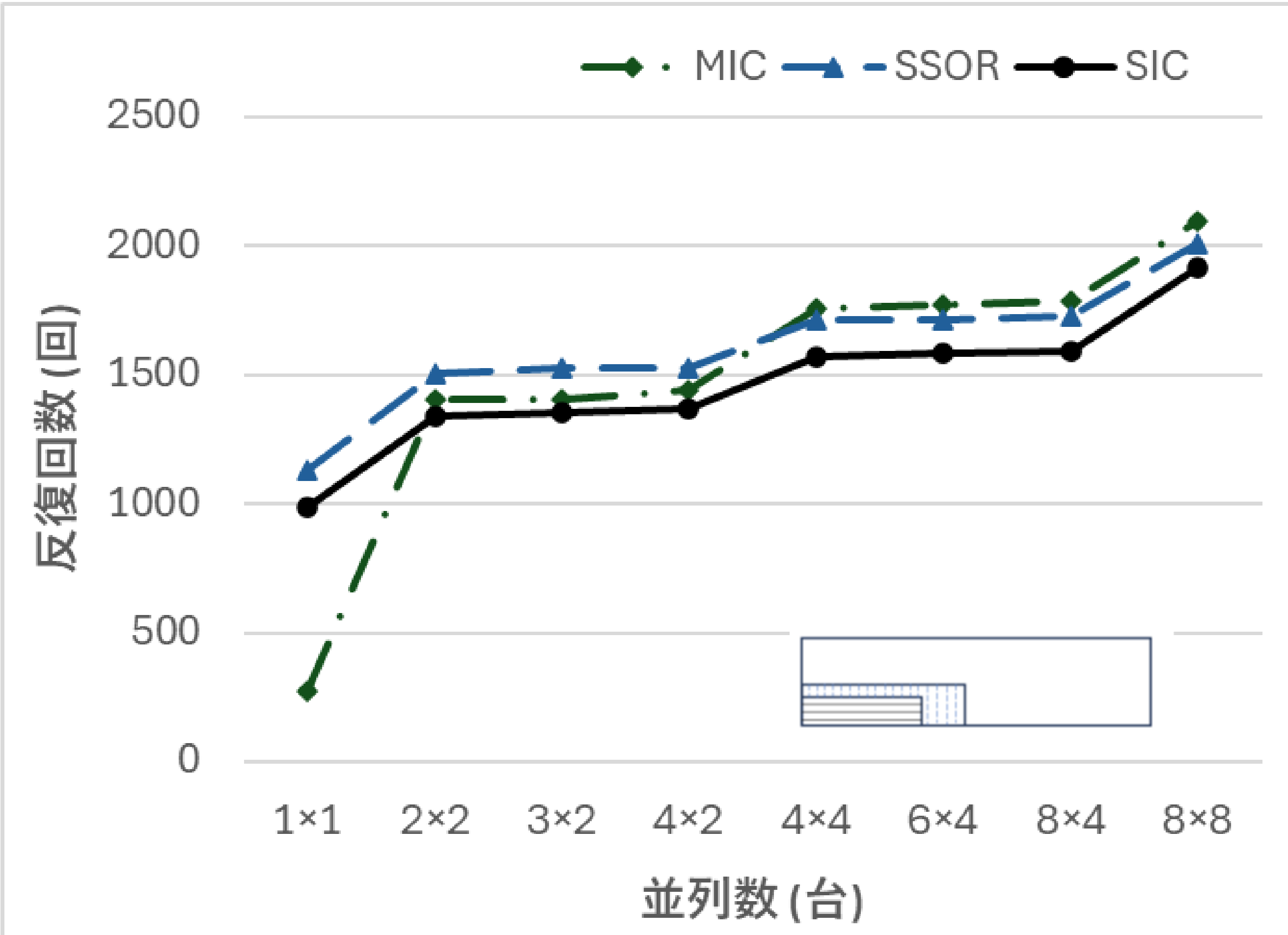
- ω=1.48,　γ=1.94

4.5 収束比較(3000×2000分割, M2A)



- ω=1.48,　γ=1.94

4.6 収束比較(3000×2000分割, M2D)



- ω=1.48,　γ=1.94

5 パラメータ（ω, γ）の影響

- SIC前処理は加速（ω）と上限（γ）パラメータ
- 以下の条件でωに対する収束回数を比較

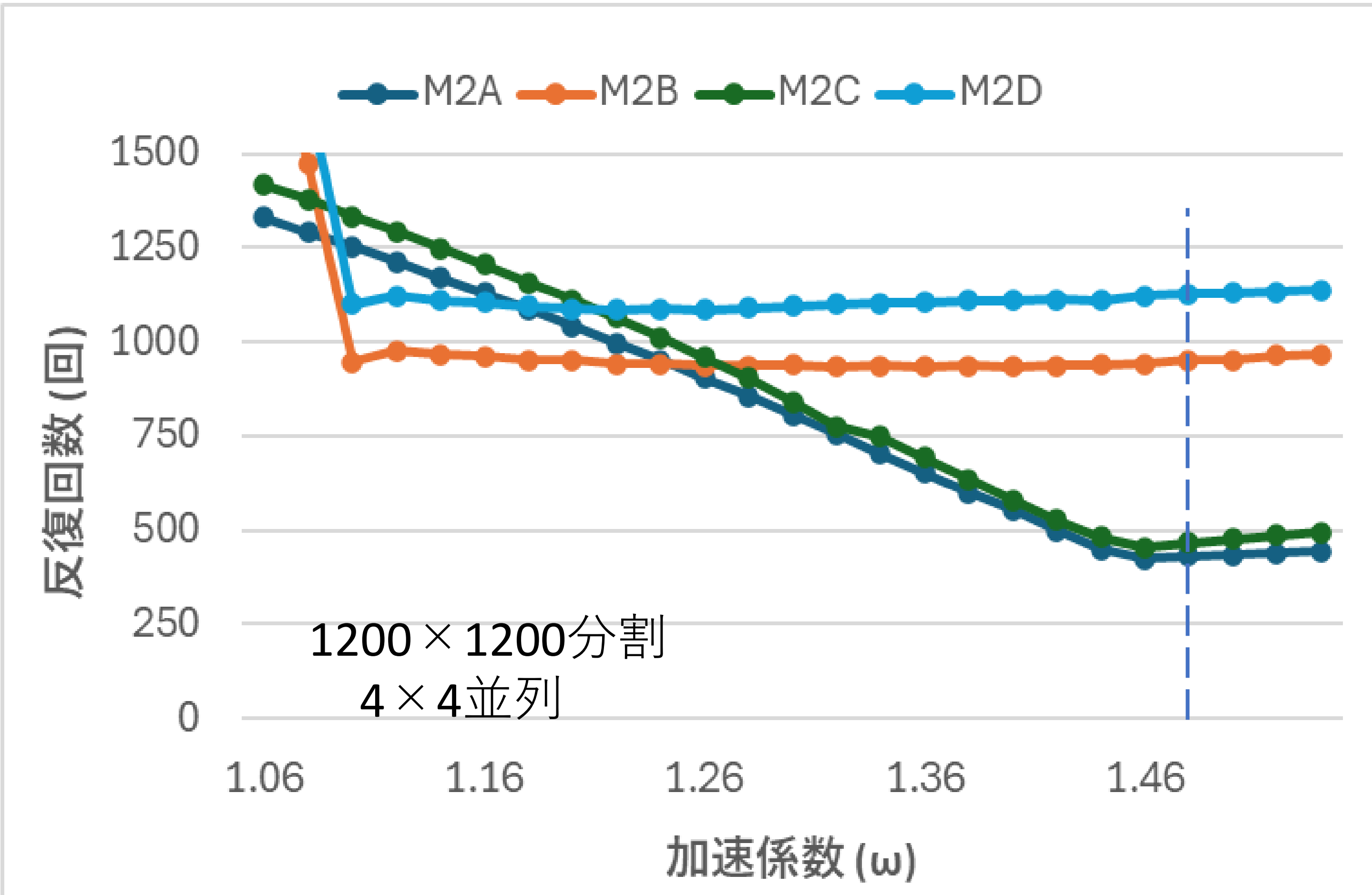
(1) 解析モデル(1200×1200分割、4×4並列)

(2) 並列数(1200×1200分割)

(3) M2A, M2D(1500×1000分割)

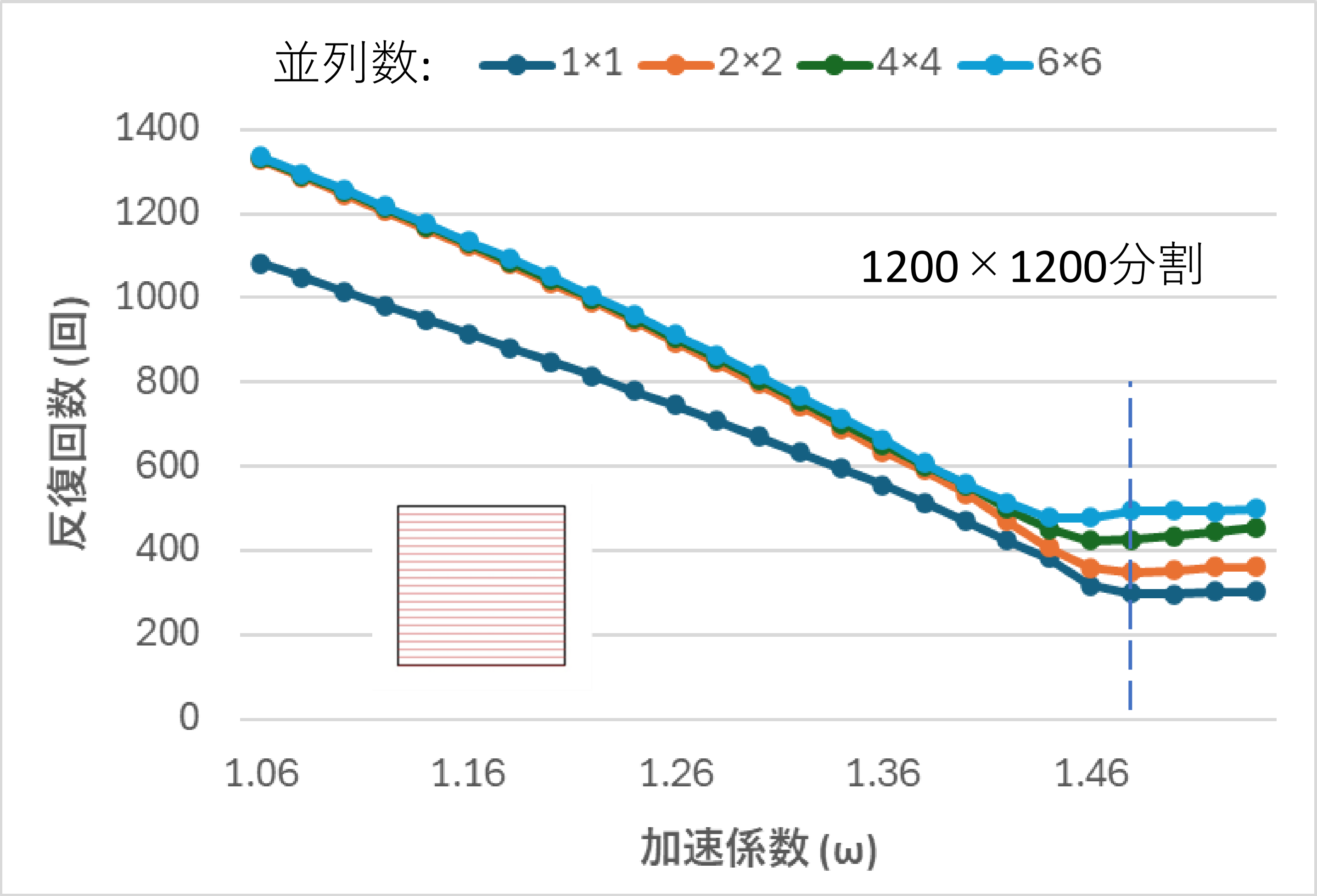
(4) 上限（γ）の影響(1.94の前後)

5.1 解析モデルへの影響



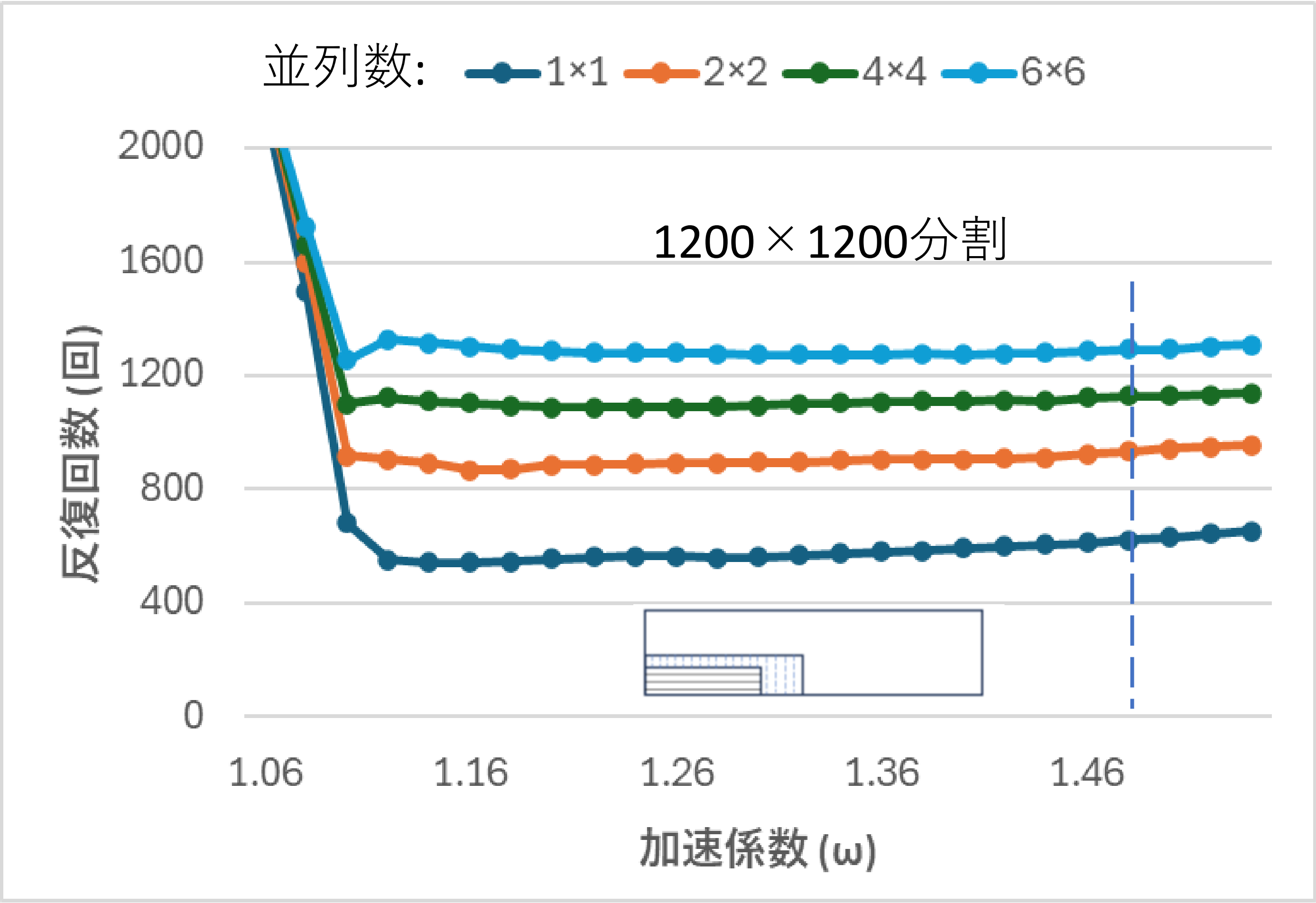
- γ=1.94

5.2 並列数への影響(M2A)



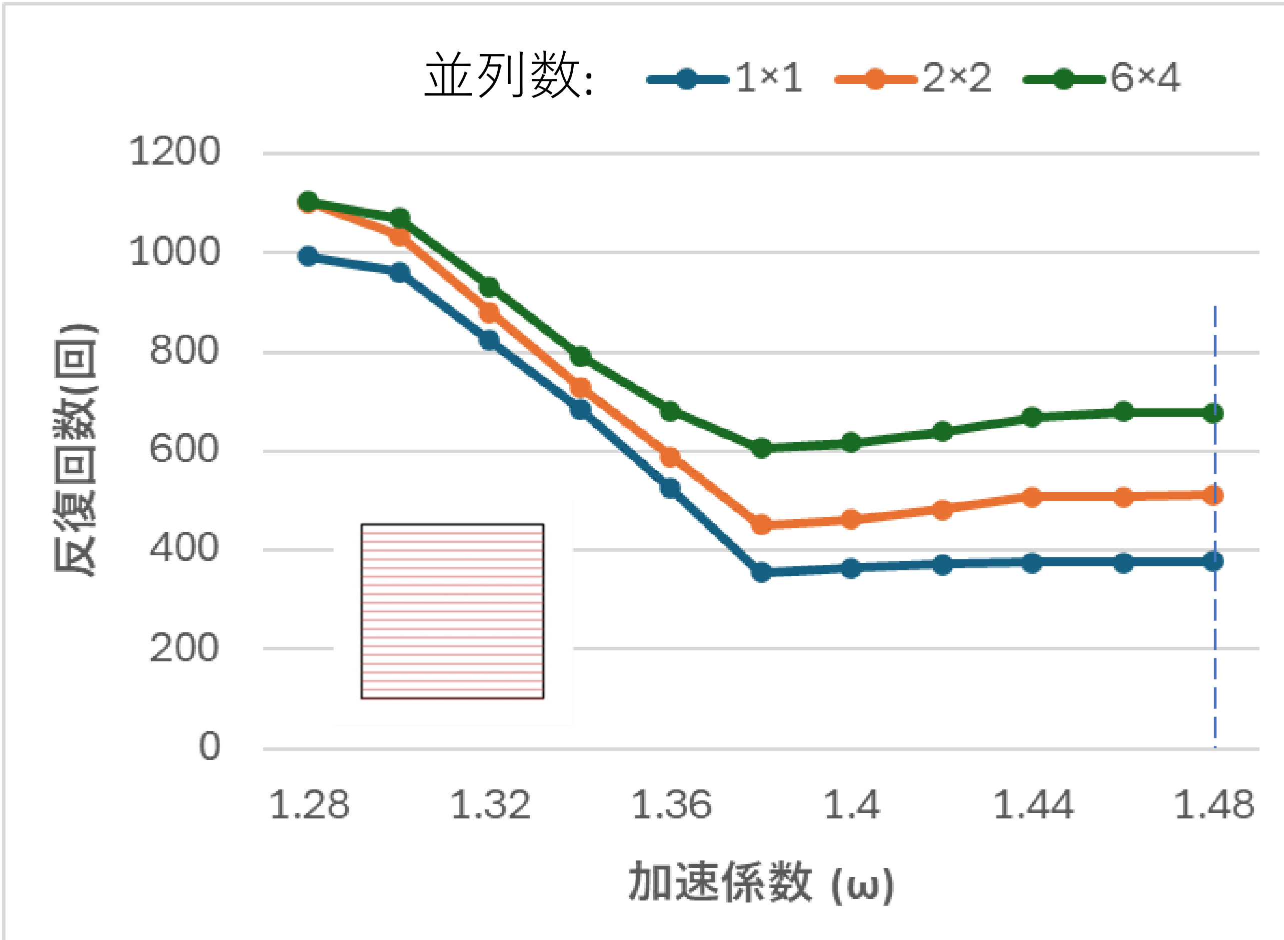
- γ=1.94

5.3 並列数への影響(M2D)



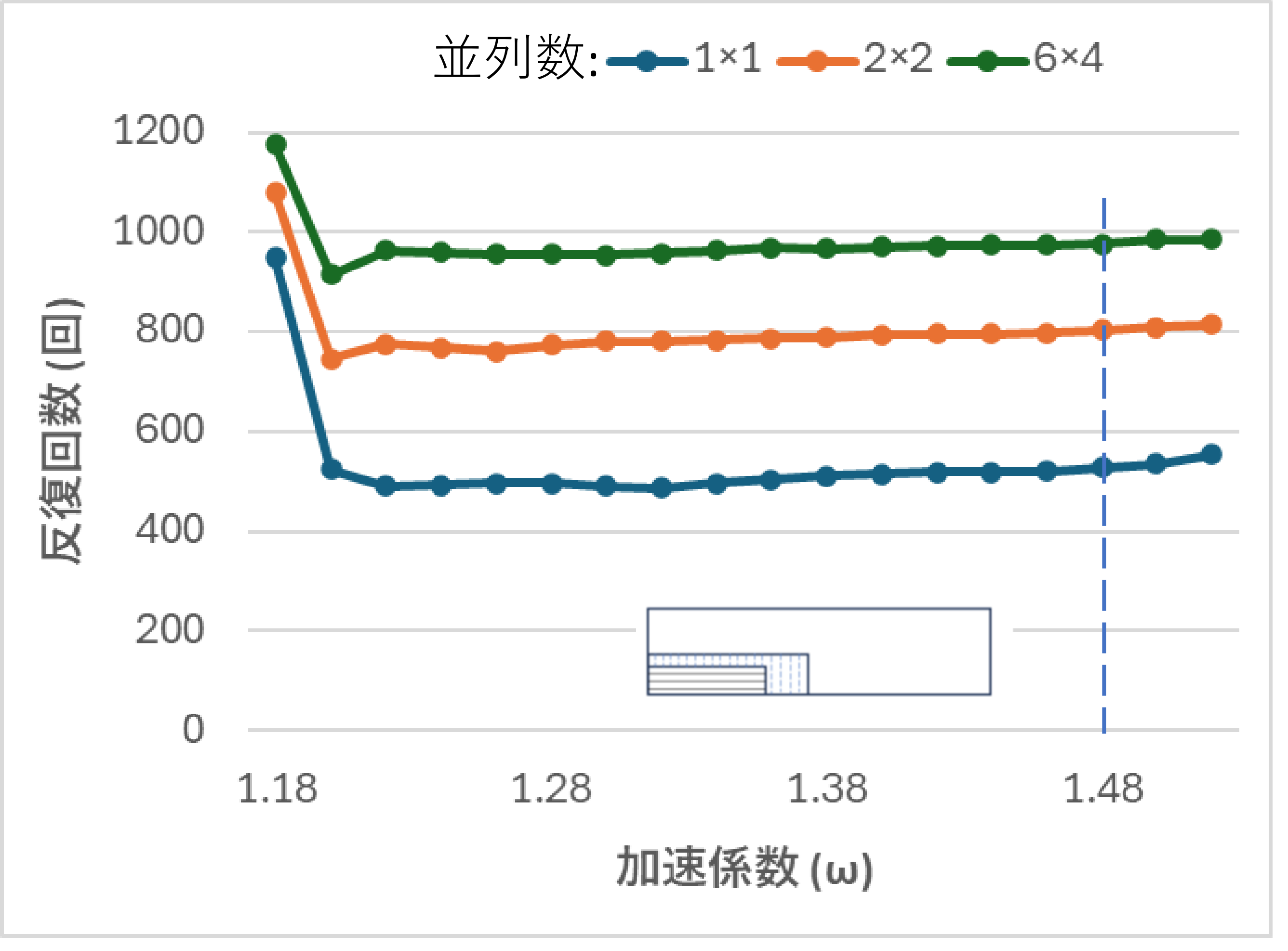
- γ=1.94

5.4 1500×1000分割(M2A)



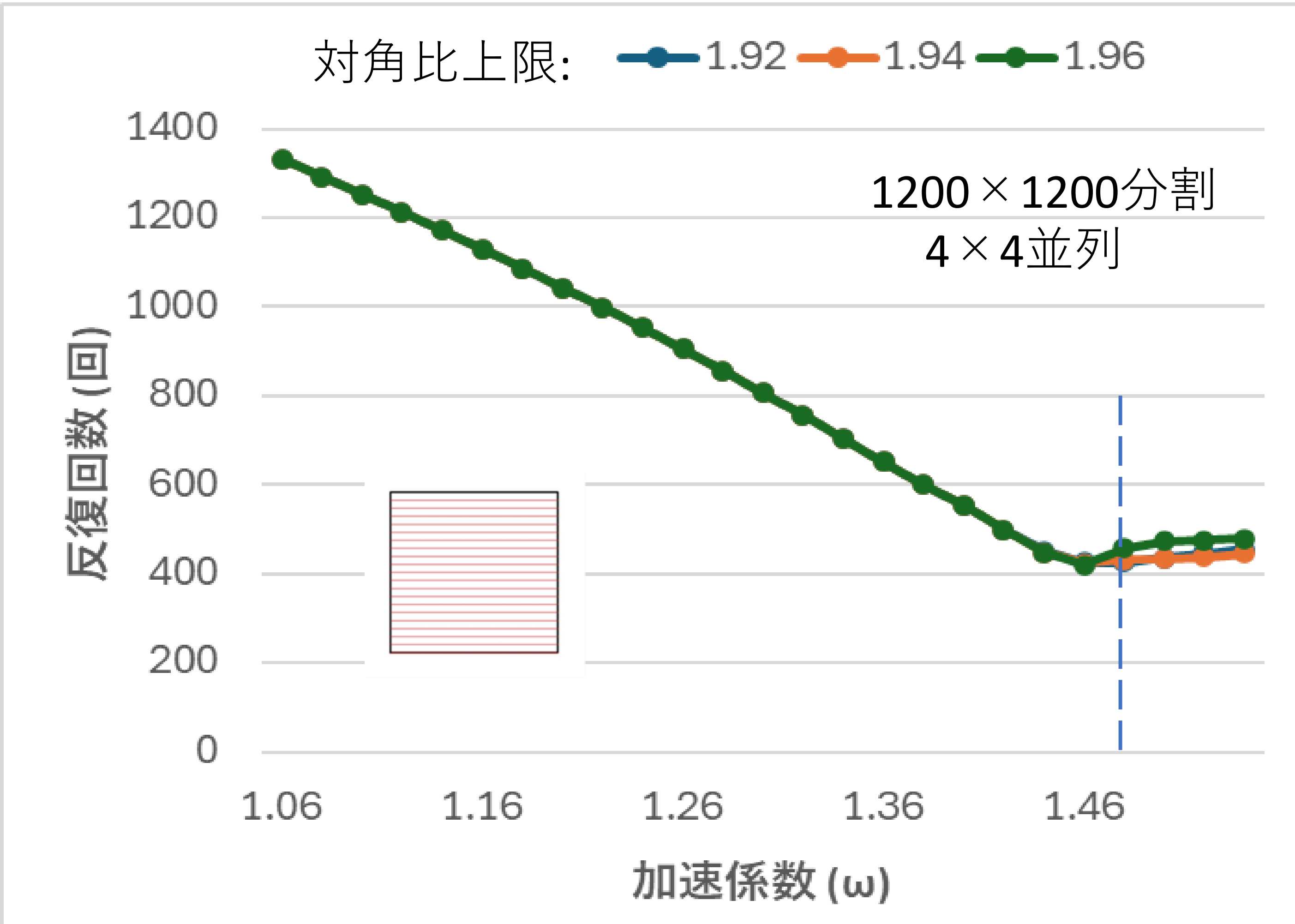
- γ=1.94

5.5 1500×1000分割(M2D)

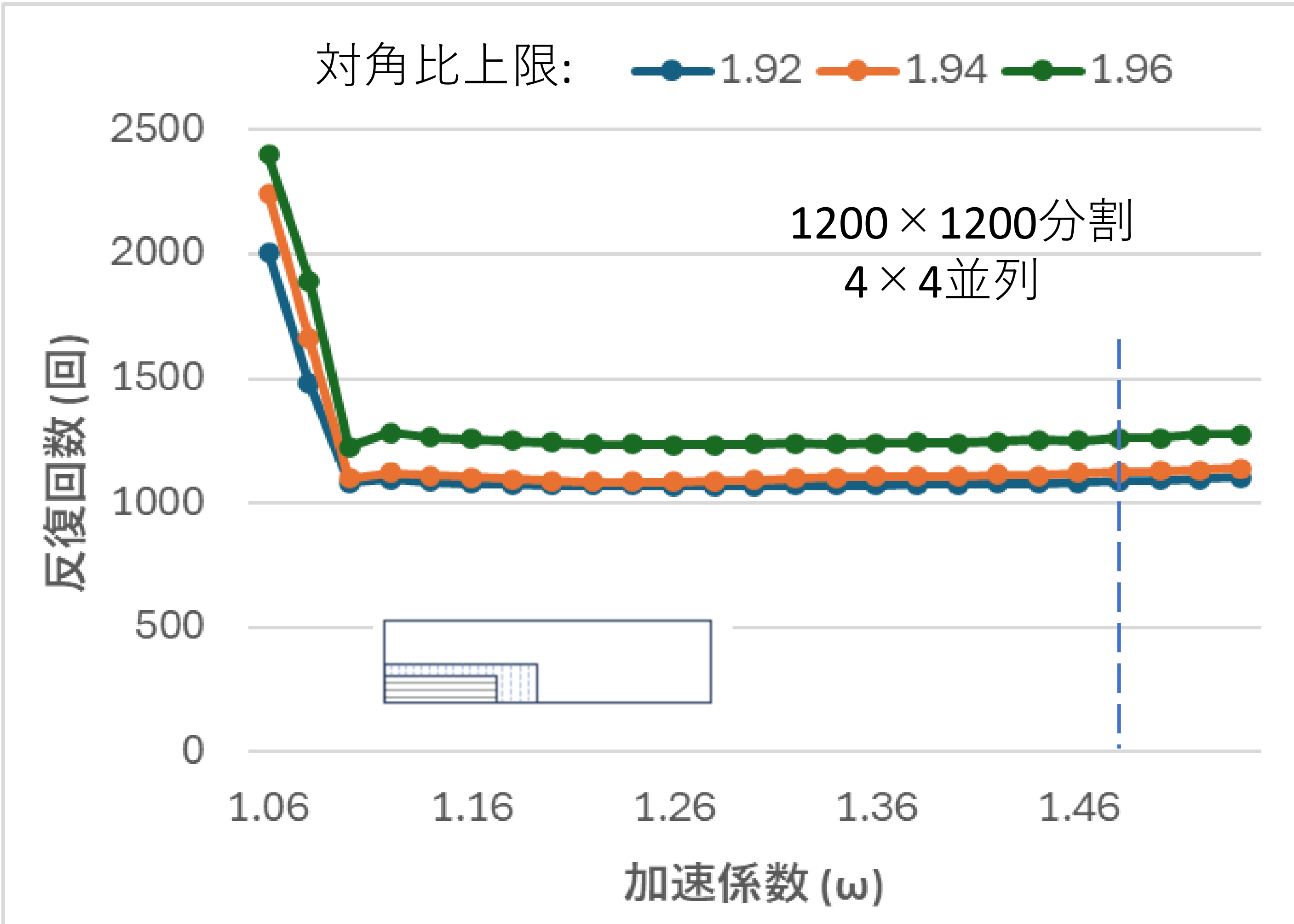


- γ=1.94

5.6 上限(γ)の影響(M2A)



5.7 上限(γ)の影響(M2D)

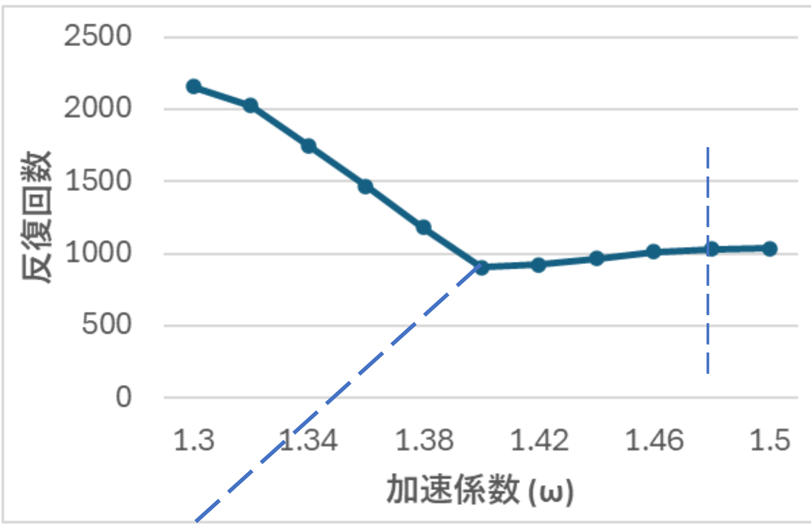


6 加速(ω)と上限(γ)の自動設定

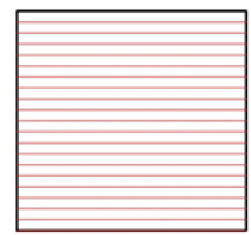
- SIC前処理は不完全LDL^T分解で下記情報
- 上限(γ)を超えた要素の比率(超過比率)
  SSOR法では得られない
- ωが少し増加で超過比率は劇的増加
- 劇変のωを使用すと反復回数が少ない
  → このωを利用(自動設定)
- ２分法で劇変のωの幅を狭める
- γは要素数(分割数/並列数)で決める

6.1 超過比率劇変例(M2A)

- 加速係数ωでγ(1.94)を超える要素の比率



3000×2000分割
4×4分割

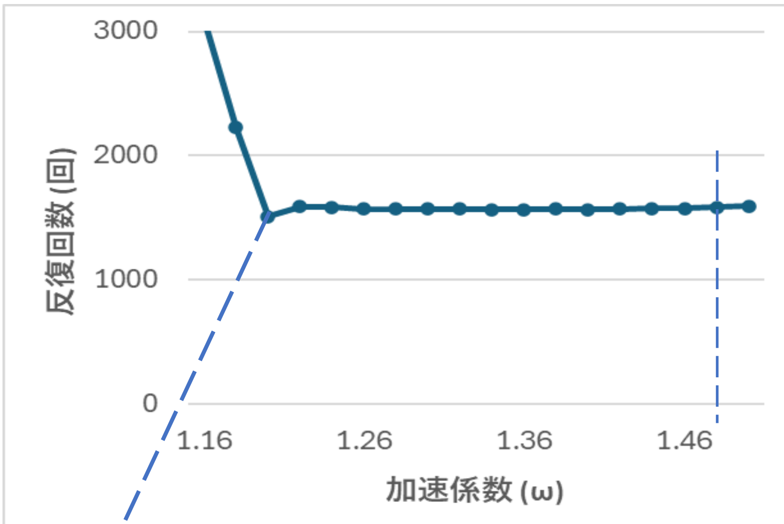


ω=1.398 反復=928 比率=0.00
ω=1.399 反復=916 比率=0.00
ω=1.400 反復=905 比率=0.99 ← 採用
ω=1.401 反復=905 比率=0.99

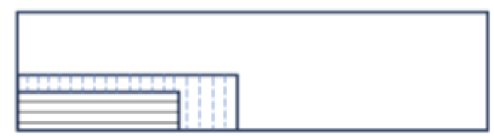
ω×対角/(対角−IC処理)
がγを超える要素比率

6.2 超過比率劇変例(M2D)

- 加速係数ωでγ(1.94)を超える要素の比率



3000×2000分割
4×4並列



ω×対角/(対角-IC処理)
がγを超える要素比率

ω=1.201 反復=1495 比率=0.00
ω=1.202 反復=1488 比率=0.00
ω=1.203 反復=1487 比率=0.98 ← 採用
ω=1.204 反復=1498 比率=0.98

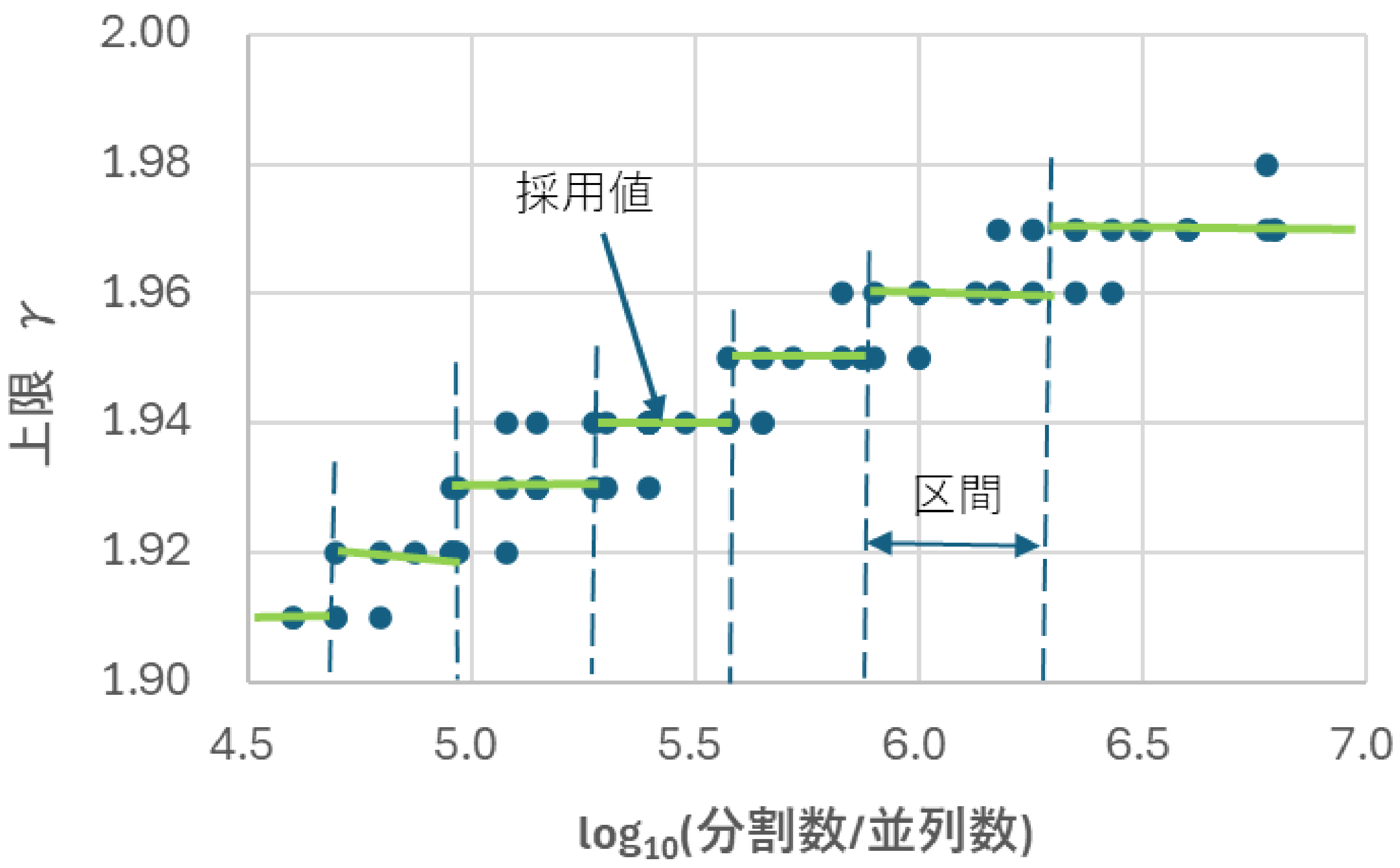
6.3 ωの自動設定方法

- 超過比率(β)付き不完全LDL^T分解を利用
- 初期ωをω=1.3、増分ε=0.15に設定
- 下記を約10回繰り返す
  (1) ωの不完全LDL^T分解でβを算出
  (2) $\beta<0.5$で$\omega=\omega+\varepsilon$、以外で$\omega=\omega-\varepsilon$
  (3) εをε/2に更新
- 最後のωと不完全分解を使用しCG法を実施
- $1.0<\omega<1.6$を探し、εは0(10^{-4})となる

6.4 収束の速いγの決定

- 要素数(分割数/並列数)とモデルを変え実行
- γ=1.9から1.98まで0.01単位で測定
- 測定はγに対応したωを自動計算し、収束が速いγを選ぶ
- $\log_{10}$(分割数/並列数)を横軸に、収束の速いγを縦軸にしたグラフを作成
- このグラフから$\log_{10}$(分割数/並列数)の区間値をに対応するγを決める

6.5 要素数と収束の速いγ



6.6 上限(γ)の設定

- $\lambda = \log_{10}$(分割数/並列数)
- λにより下記よりがγを決定

		λ	$<$	4.7	$\rightarrow$	γ	= 1.91
4.7	$<=$	λ	$<$	5.0	$\rightarrow$	γ	= 1.92
5.0	$<=$	λ	$<$	5.3	$\rightarrow$	γ	= 1.93
5.3	$<=$	λ	$<$	5.6	$\rightarrow$	γ	= 1.94
5.6	$<=$	λ	$<$	5.9	$\rightarrow$	γ	= 1.95
5.9	$<=$	λ	$<$	6.3	$\rightarrow$	γ	= 1.96
6.3	$<=$	λ			$\rightarrow$	γ	= 1.97

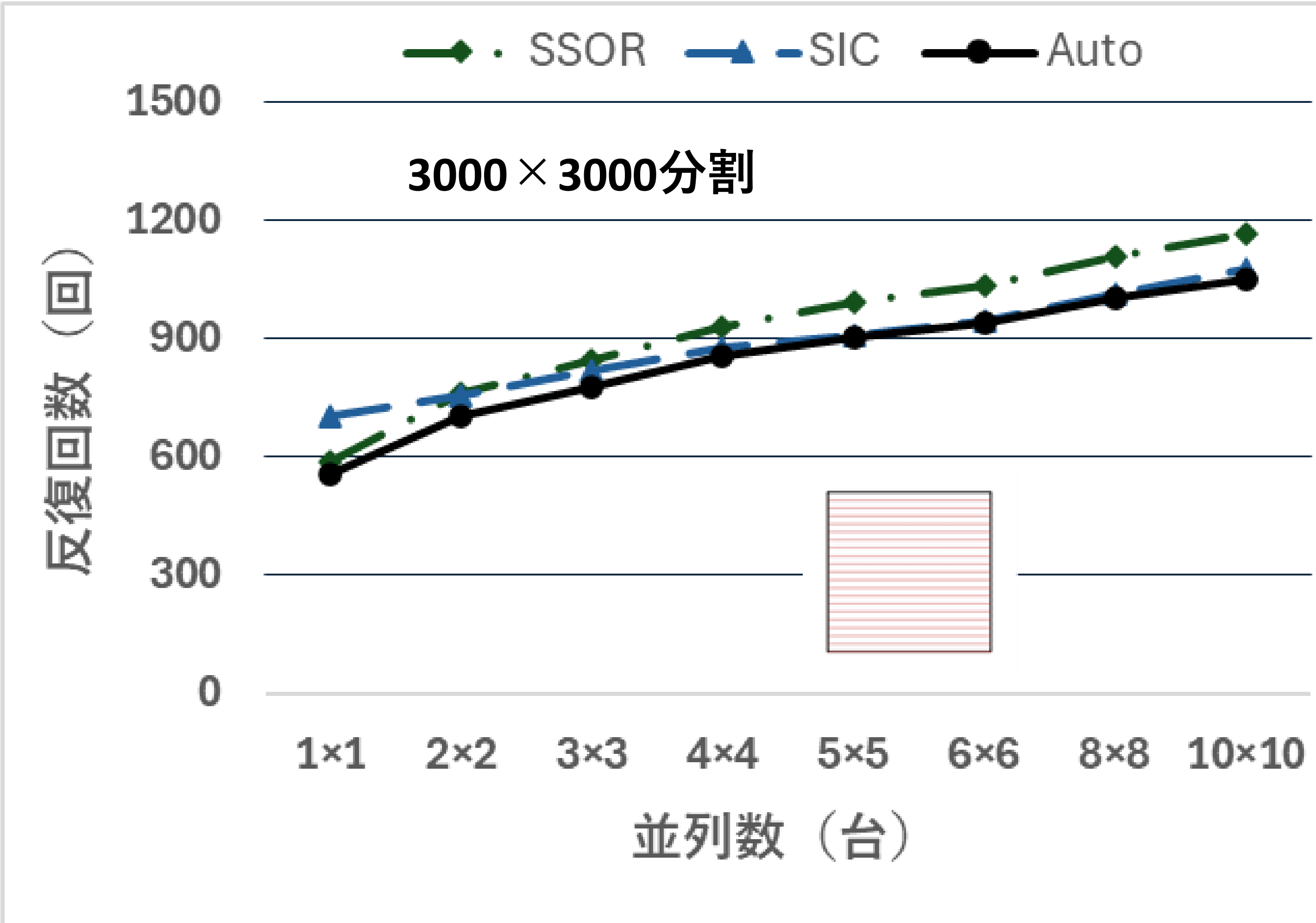
7 自動パラメータ(ω, γ)の効果

- CG法の並列用前処理法を検討
- ICに加速(ω)と上限(γ)を付けた前処理を提案
- 2次元中心差分法で収束性を評価した
- SSOR、SIC(固定)、SIC(自動:Auto表示)で評価
- SSOR前処理の最適ωよりAutoは収束が速い
- 収束の速いω, γが自動算出できる

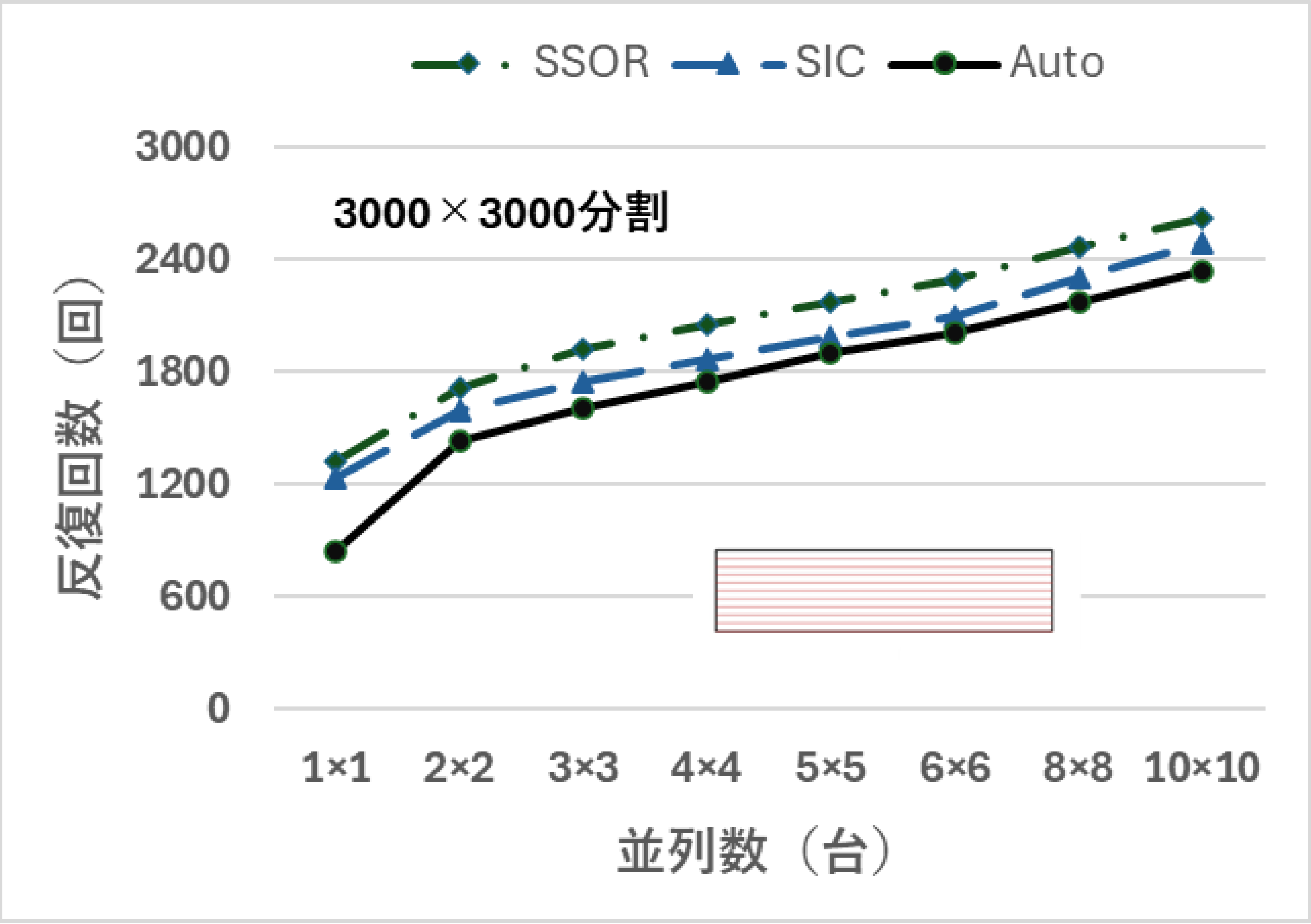
  ωは不完全LDL^Tの情報で決める

  γは$\log$(分割数/並列数)で決める

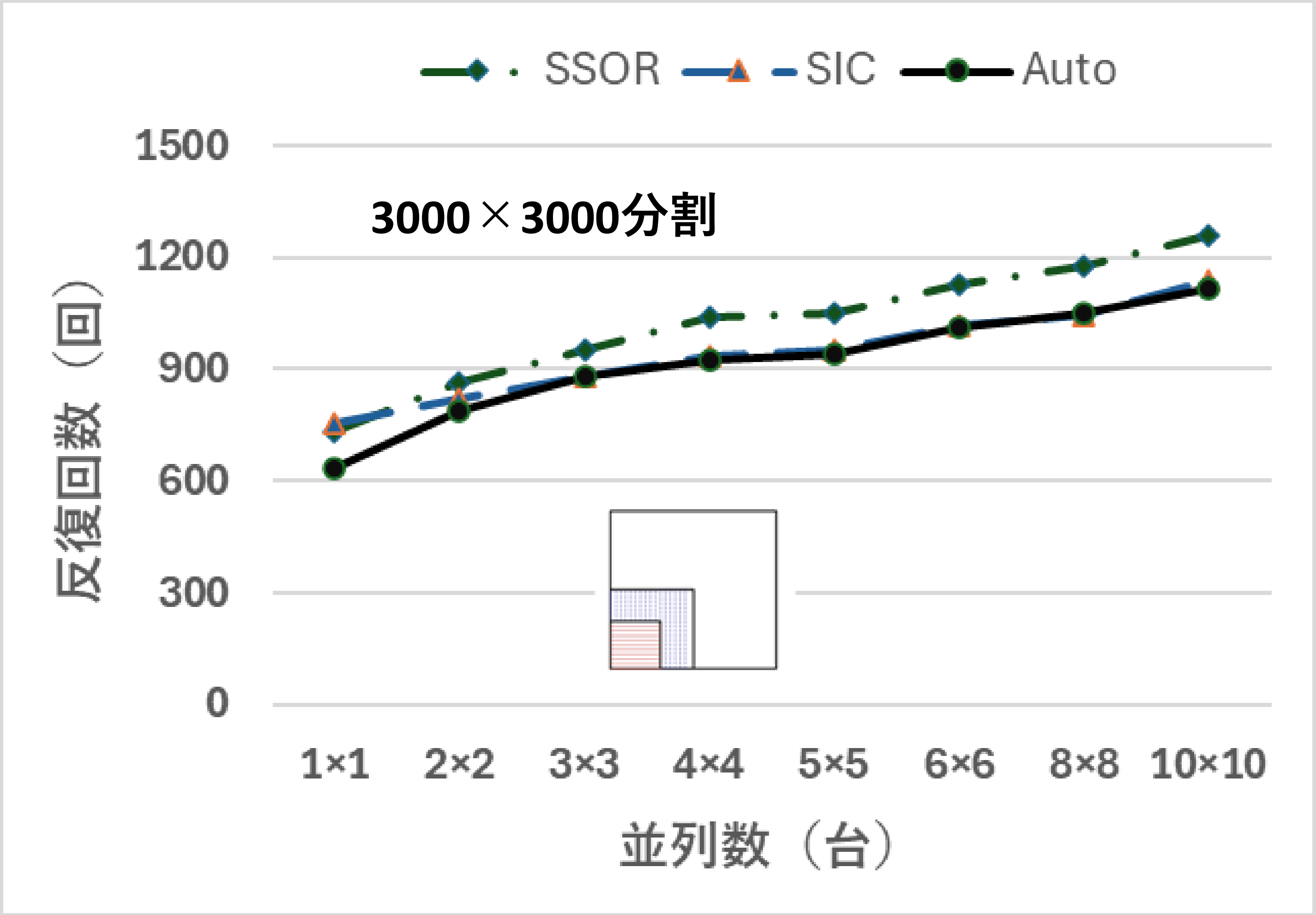
7.1 収束比較(並列数、M2A)



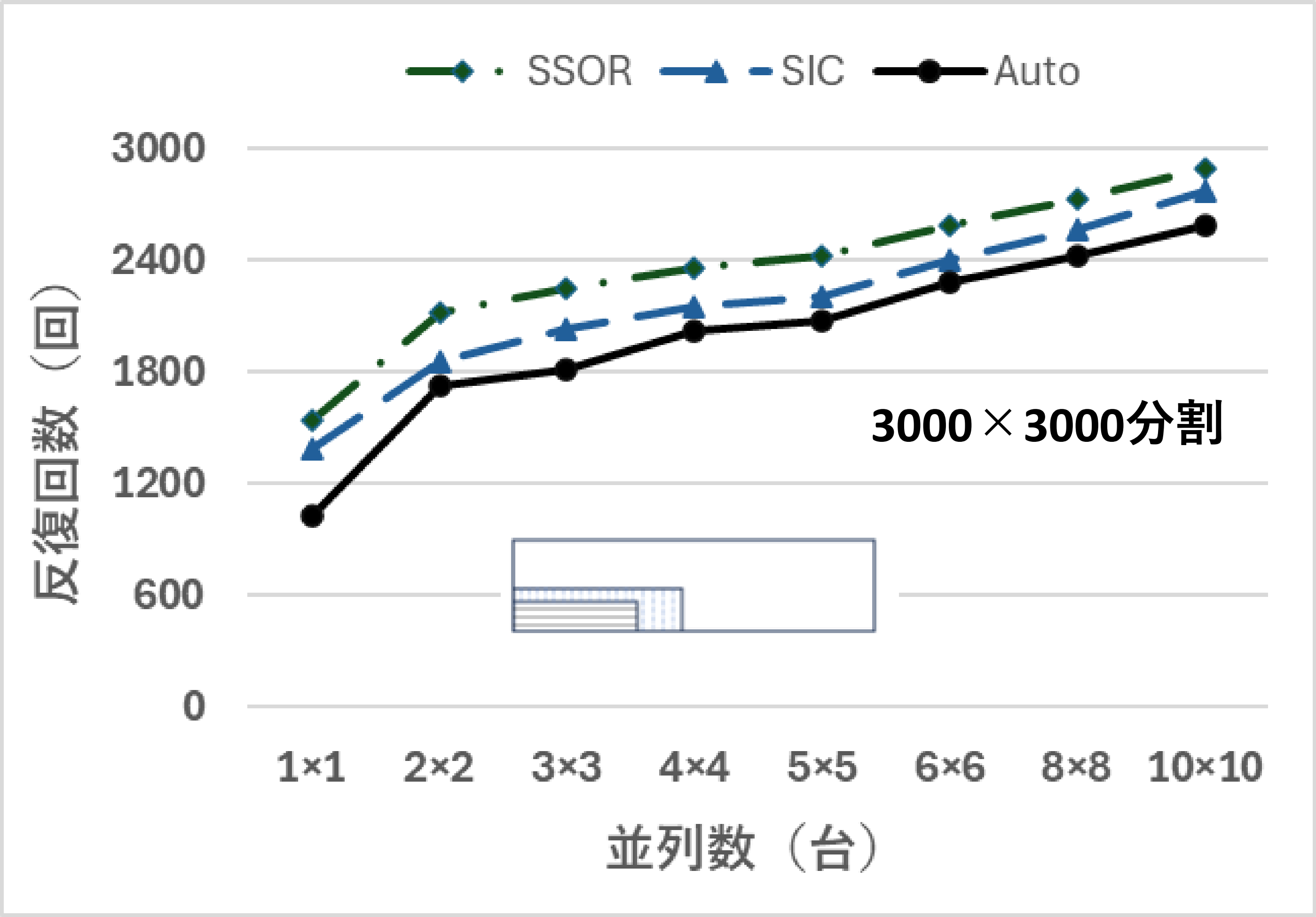
7.2 収束比較(並列数、M2B)



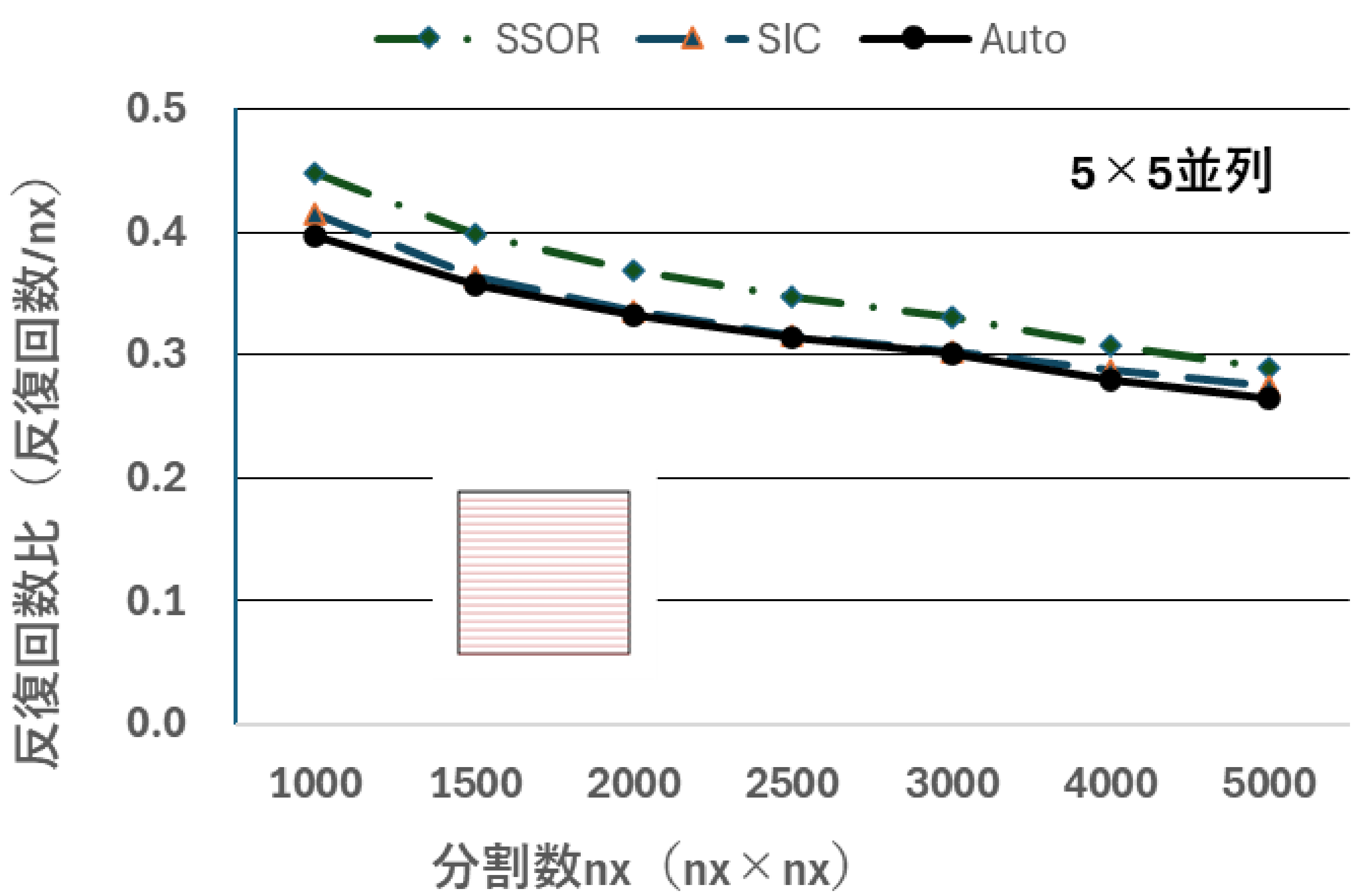
7.3 収束比較(並列数、M2C)



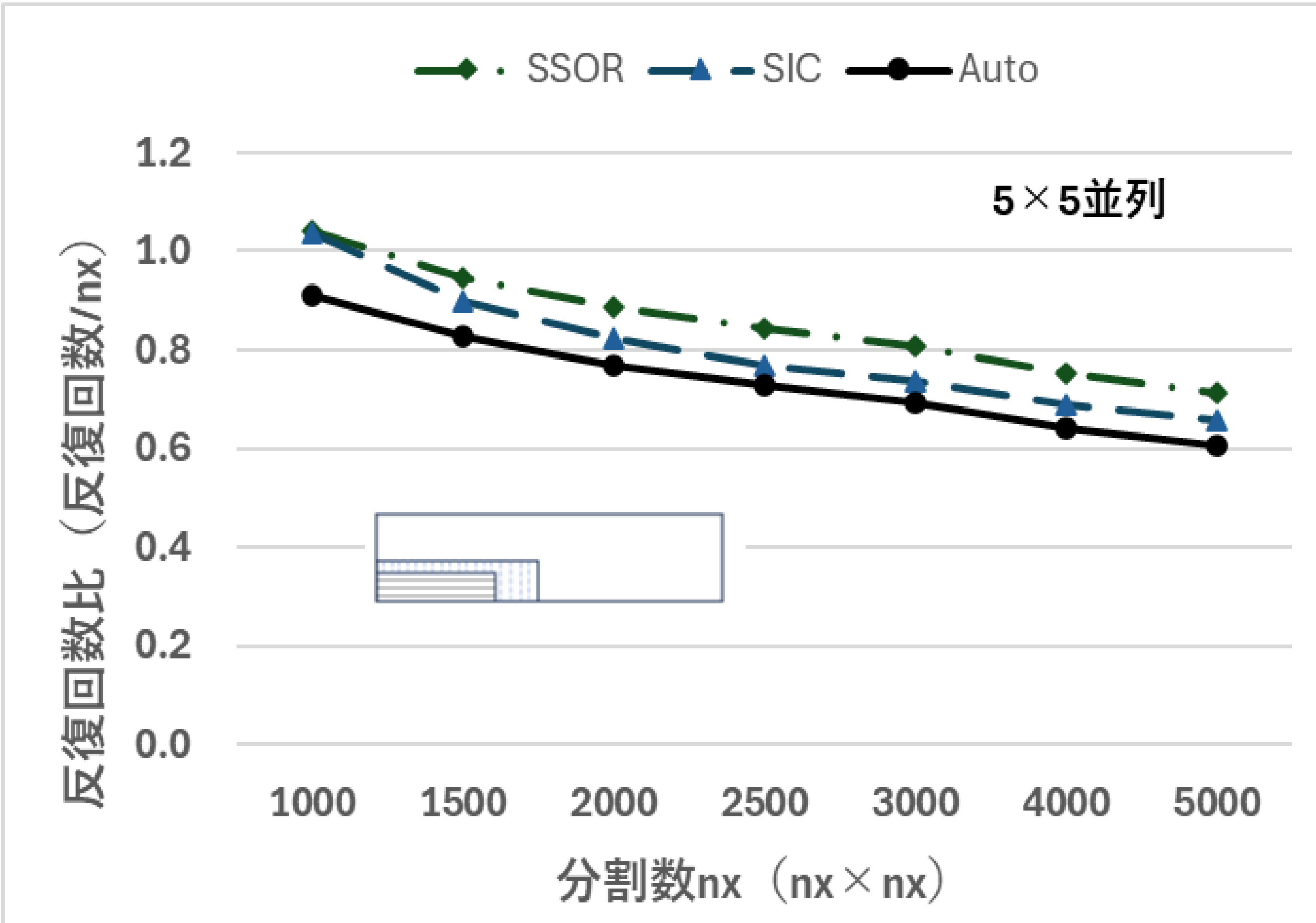
7.4 収束比較(並列数、M2D)



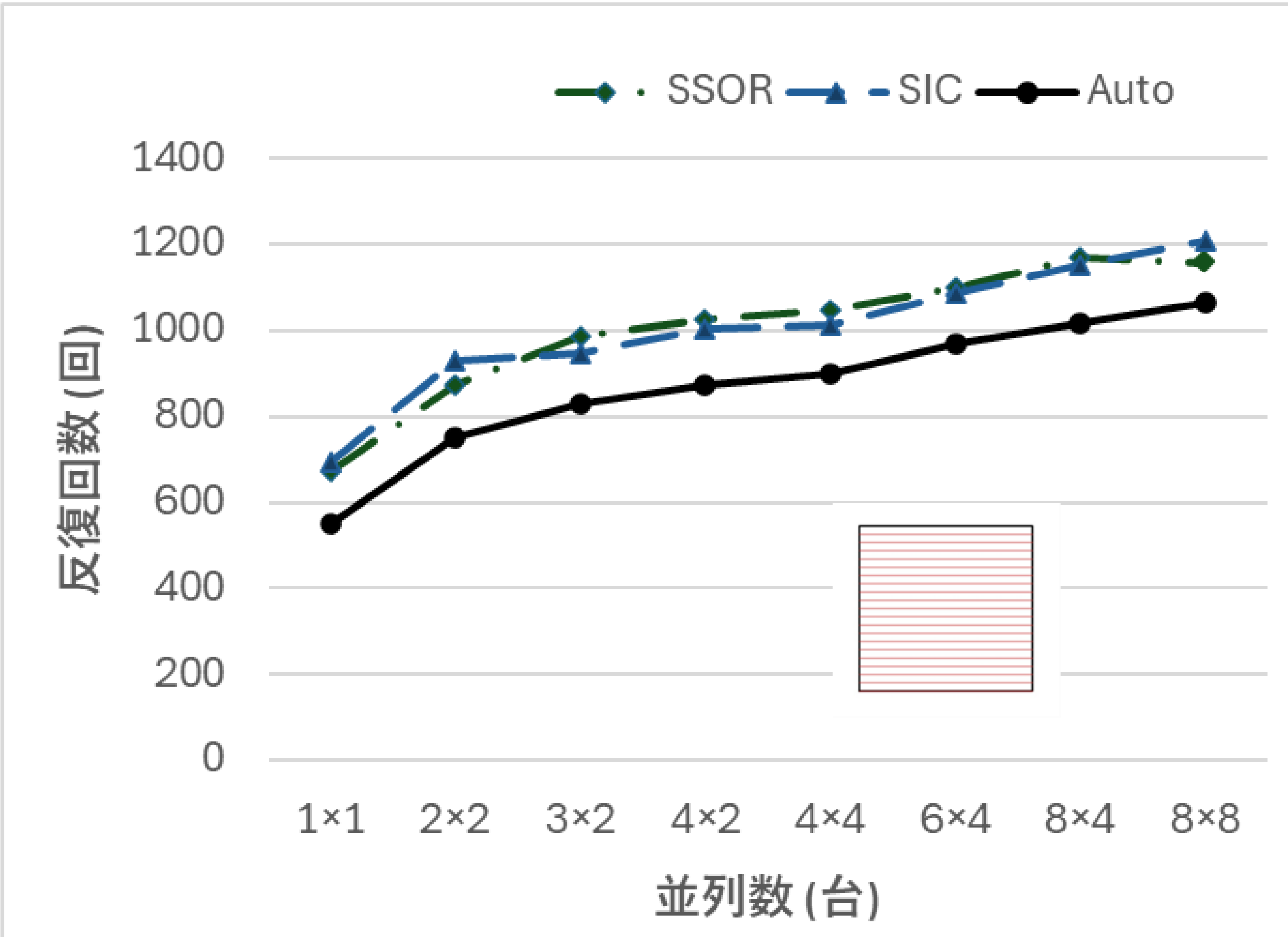
7.5 収束比較(分割数、M2A)



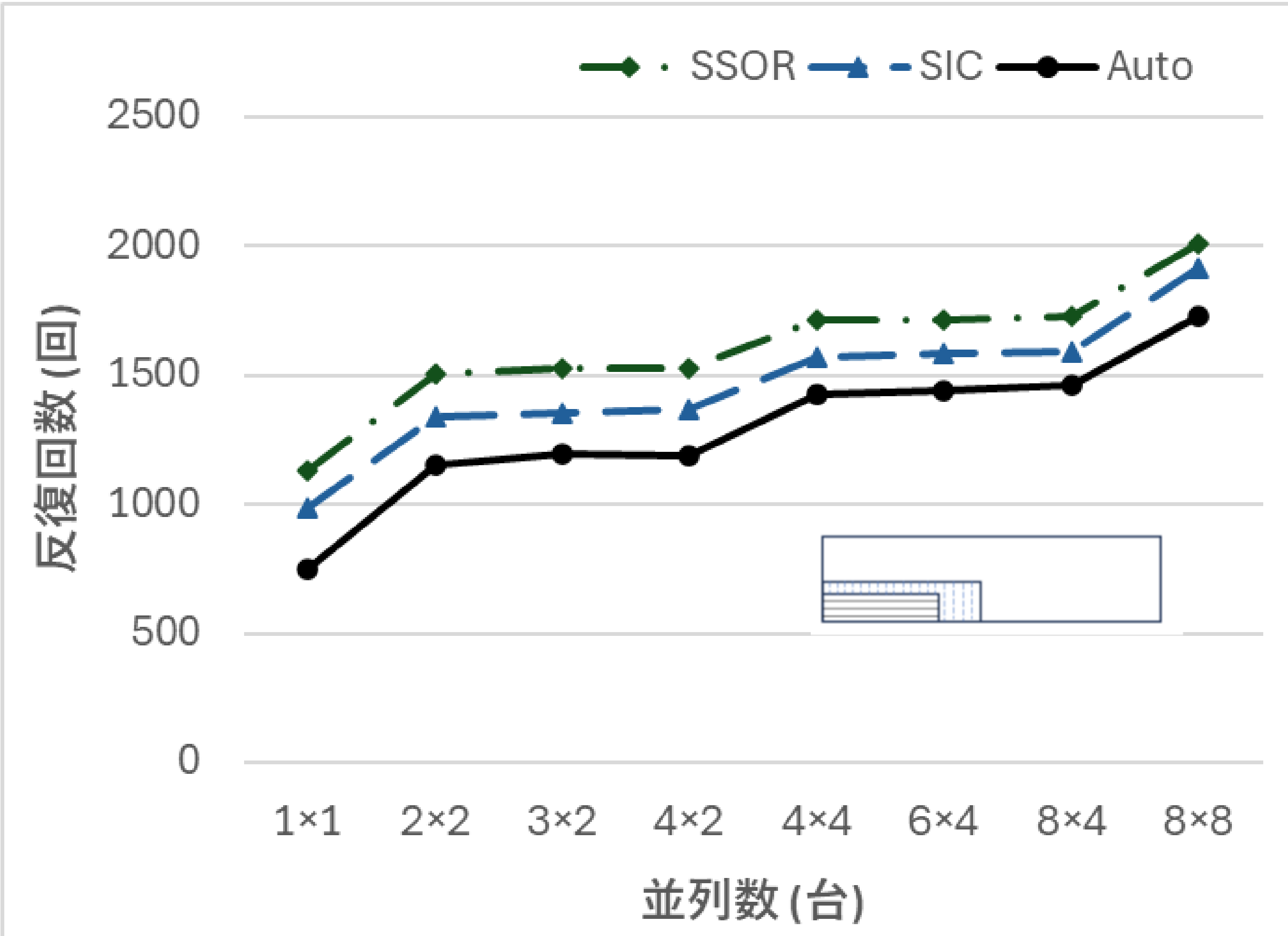
7.6 収束比較(分割数、M2D)



7.7 収束比較(3000×2000分割、M2A)



7.8 収束比較(3000×2000分割、M2D)



8 おわりに

- CG法の並列用前処理法を検討
- ICに加速(ω)と上限(γ)を付けた前処理を提案
- 2次元中心差分法で収束性を評価した
- ω=1.48, γ=1.94でSSOR前処より少し良い
- 収束の速いω, γが自動算出できた
- 拡散方程式(差分法)の反復解法は下記推奨

スカラ計算 → **MICCG法(規模の対数でα算出)**

並列計算 → **提案解法(SICCG)**

ωは自動算出、γは規模で算出