Simulation and Scientific Computing – 2 (SS 2016)

Multigrid Solver

Team :-

BleedBlue

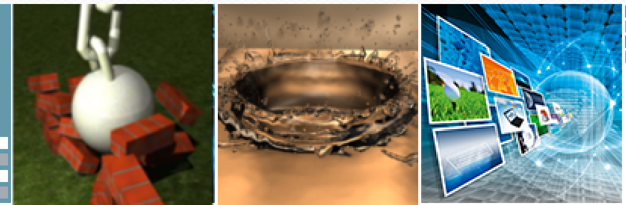
Paras Kumar

Raju Ram

Iniyan Kalaimani

Performance Challenge

Informatik 10
System Simulation



FRIEDRICH-ALEXANDER UNIVERSITÄT ERLANGEN-NÜRNBERG
TECHNISCHE FAKULTÄT

- Startseite
- Lehre
- Forschung
- Veröffentlichungen
- Mitarbeiter
- Media
- Über uns

Sie sind hier: Startseite

SiWiR Performance Challenge

7

Group	wallclock Time	L2 Norm	Date	Host	Parallelism
VCG	0,090138	8,2214282155e-05	3. Juni 2016	i10hpc2	OpenMP
so47pefe	0,095803	8,53890912647e-05	3. Juni 2016	i10hpc2	OpenMP
JETi	0,263078	9,17553361908e-05	2. Juni 2016	i10hpc2	OpenMP
im50ibaq	0,392847	9,1766419284e-05	31. Mai 2016	i10hpc2	OpenMP
NeEmHaFl	0,404379	8,90789845053e-05	1. Juni 2016	i10hpc2	OpenMP
DeLuPhSuMaRi	0,418192	9,1750647404e-05	2. Juni 2016	i10hpc2	OpenMP
BleedBlue	0,459698	1,94895388093e-05	3. Juni 2016	i10hpc2	Serial
bu43jazu	0,486754	9,17688787287e-05	31. Mai 2016	i10hpc2	OpenMP
ADG	0,618446	9,17691397429e-05	31. Mai 2016	i10hpc2	OpenMP
teamAlias	0,697215	9,18017944771e-05	3. Juni 2016	i10hpc2	OpenMP
IndiaWale	0,879014	9,17928448577e-05	3. Juni 2016	i10hpc2	OpenMP
Simulators	0,894693	9,18200297507e-05	5. Juni 2016	i10hpc2	OpenMP
aladeen	1,66087	6,69737633681e-05	2. Juni 2016	i10hpc2	OpenMP
Cassini	1,95788	9,17239791148e-05	1. Juni 2016	i10hpc2	OpenMP
Bhai	2,02865	9,17239791148e-05	1. Juni 2016	i10hpc2	OpenMP

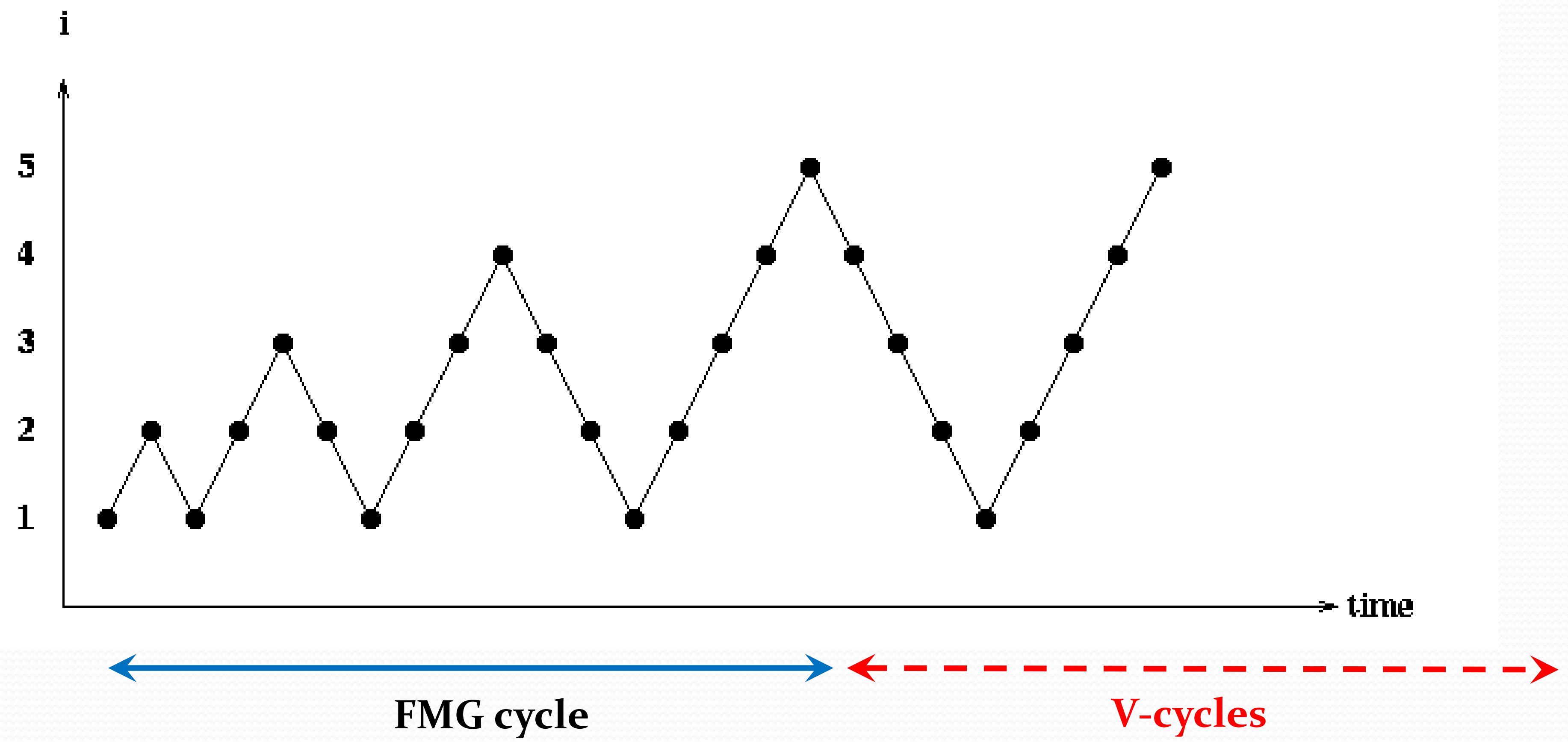
Optimizations

- Implemented
  - Multigrid Algorithm - Full Multigrid
  - Smoother - SOR solver
  - Parallelization - OpenMP

- We also tried SSE Vectorization !

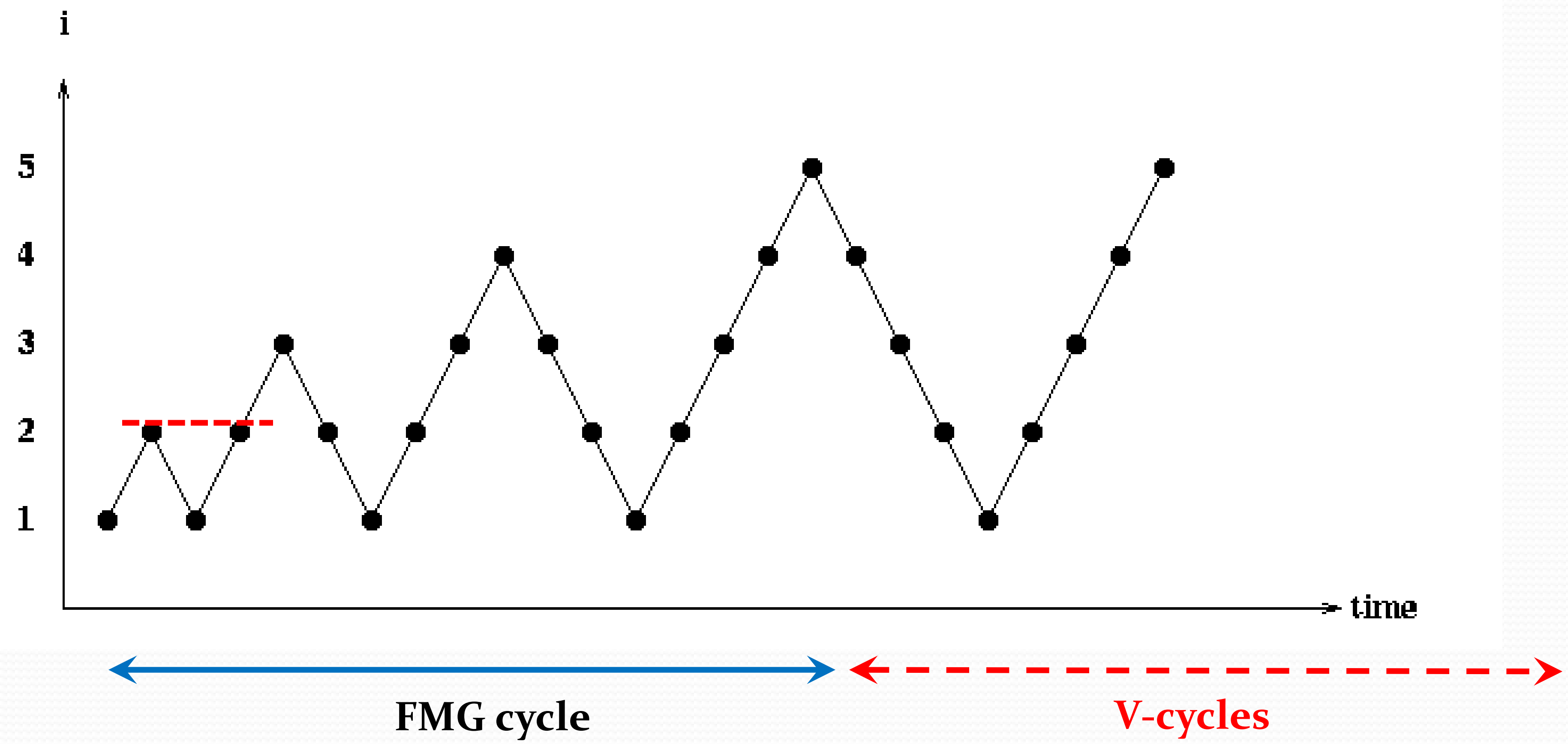
Full Multigrid cycle

- Coarse grid approximation as a first guess to obtain a good initial approximation for Multigrid cycle



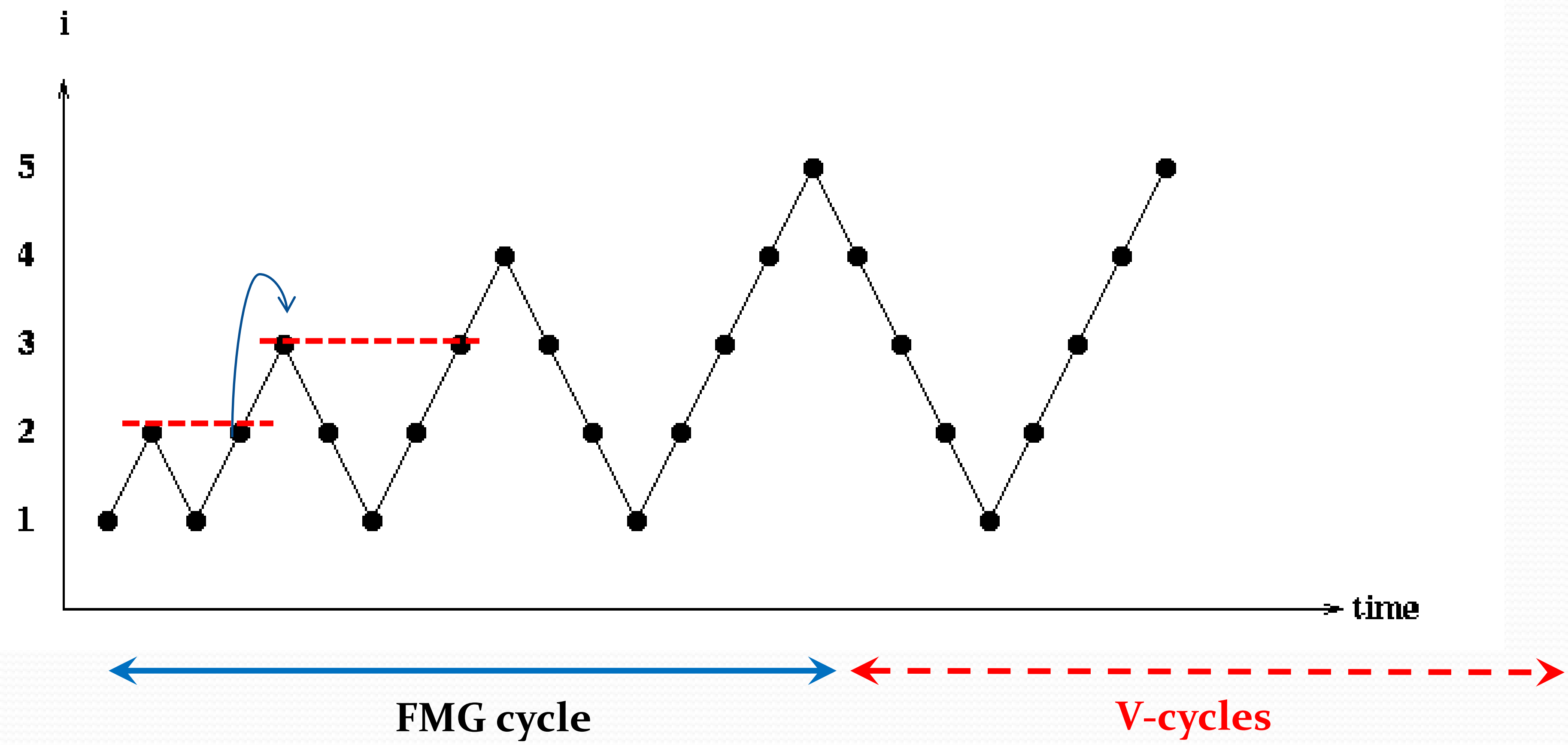
Full Multigrid cycle

- Coarse grid approximation as a first guess to obtain a good initial approximation for Multigrid cycle



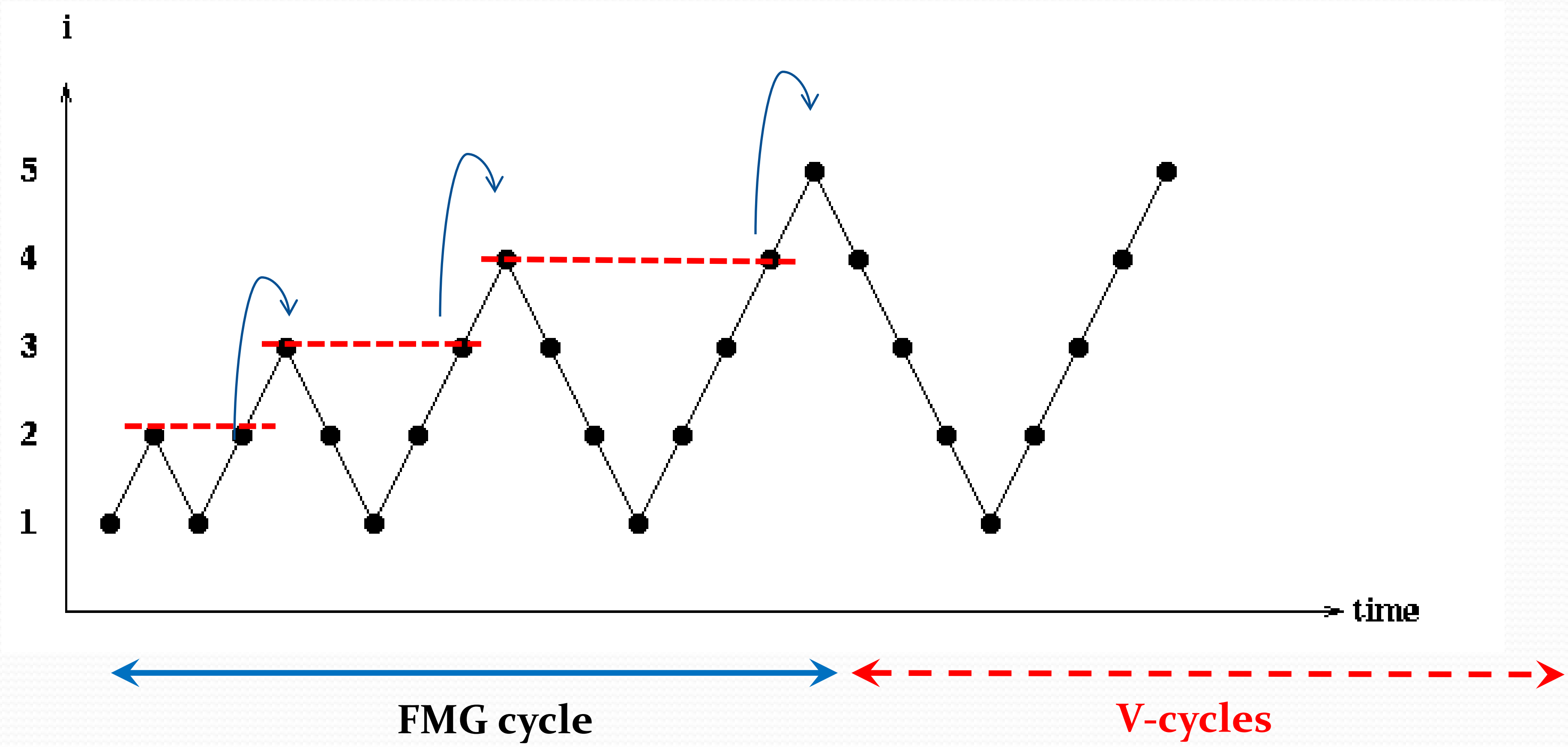
Full Multigrid cycle

- Coarse grid approximation as a first guess to obtain a good initial approximation for Multigrid cycle

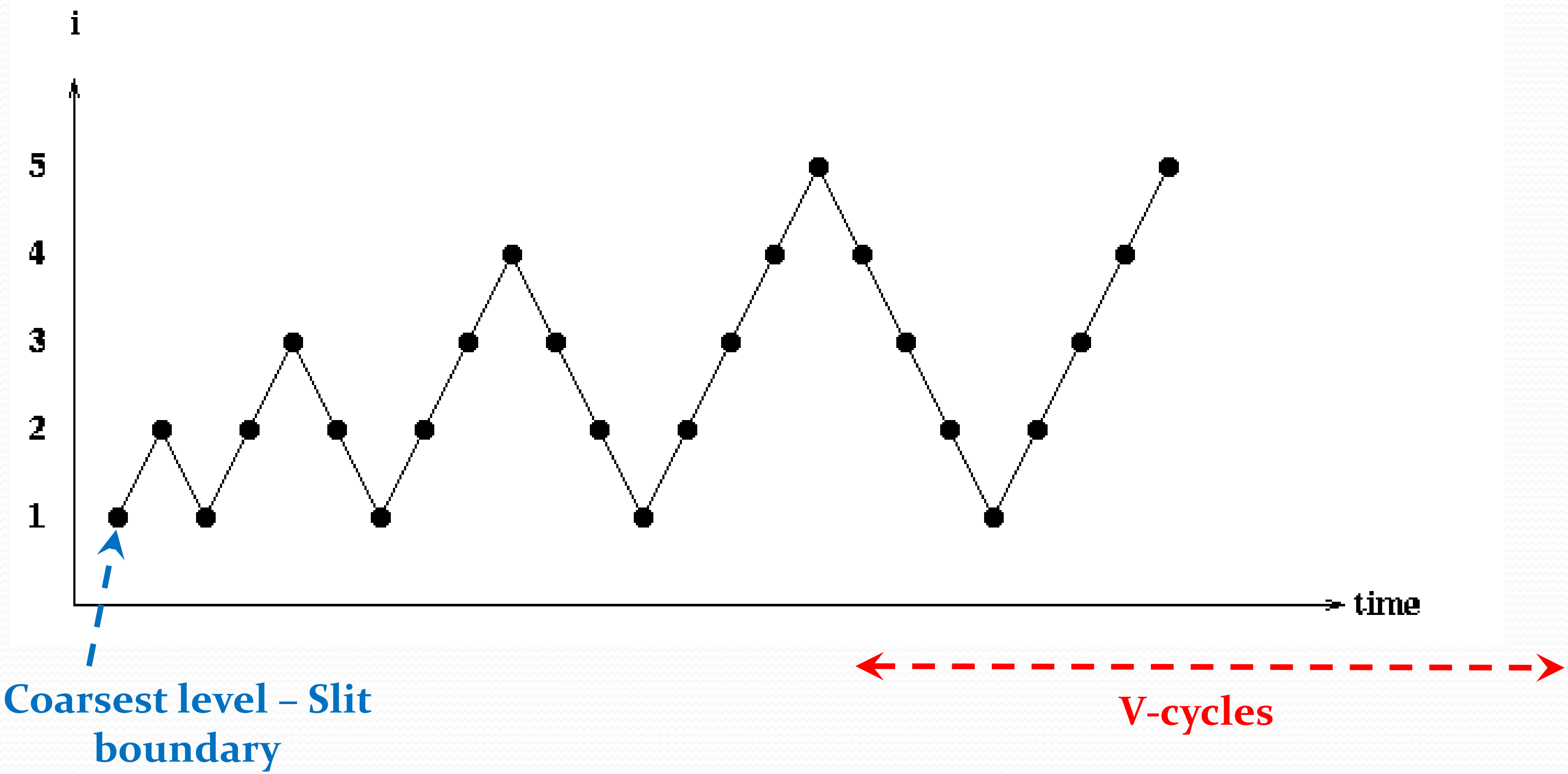


Full Multigrid cycle

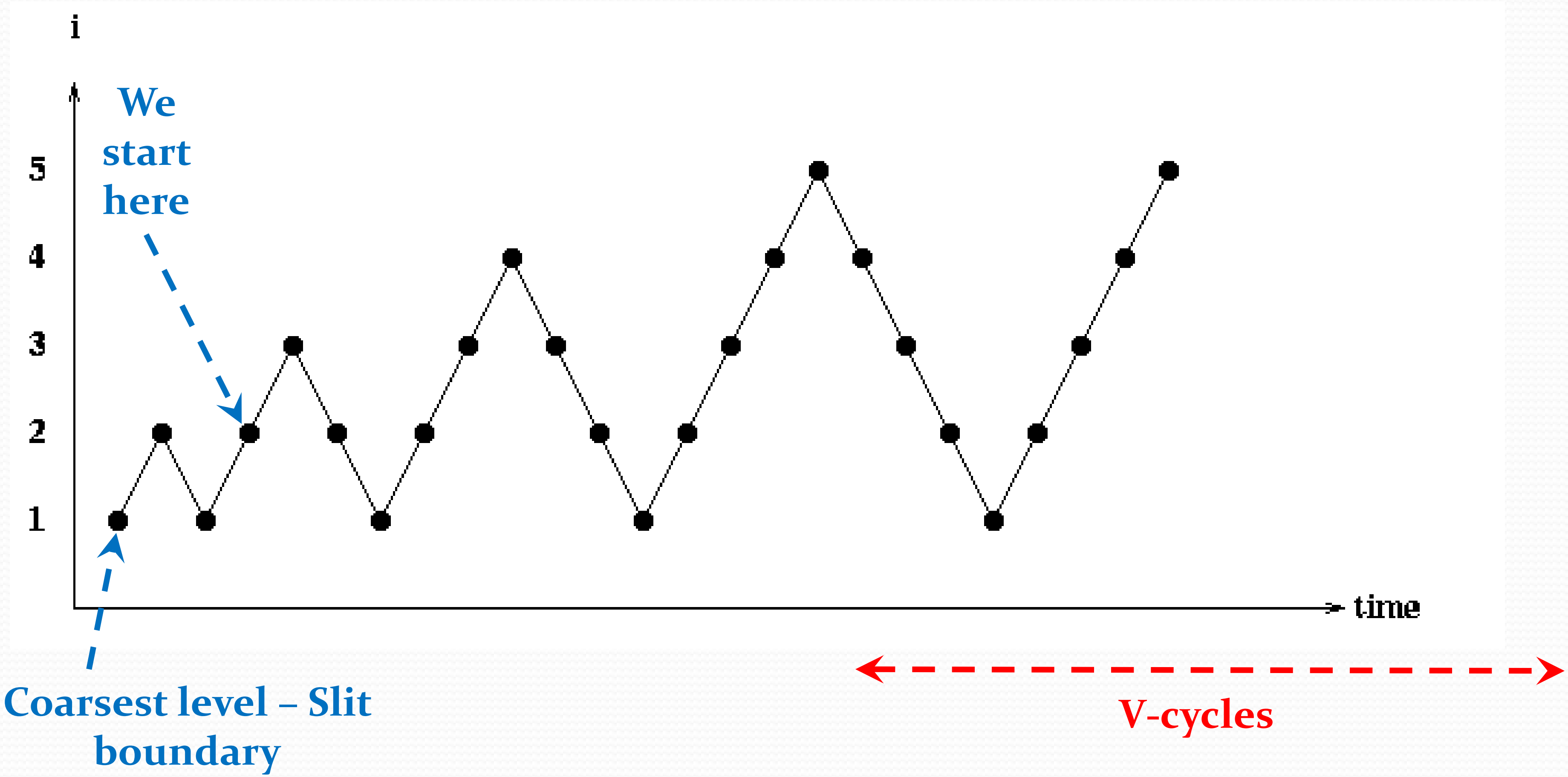
- Coarse grid approximation as a first guess to obtain a good initial approximation for Multigrid cycle



Full Multigrid cycle



Full Multigrid cycle

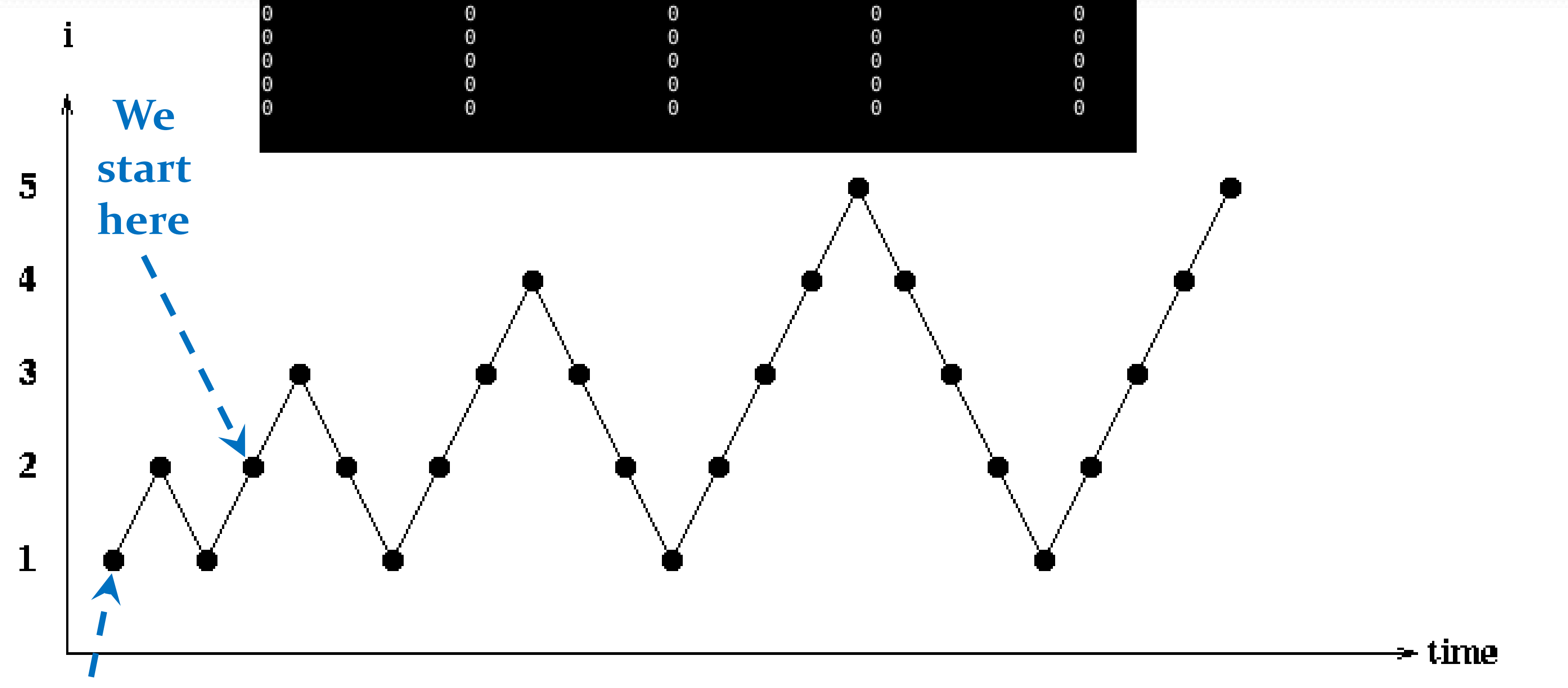


Full Multigrid cycle

```
u Before FMG

Display of u_ at level 2 during 1 v cycles


0          0          0          0          0
0          0          0          0          0
0          0          0          0          0
0          0          0          0          0
0          0          0          0          0
```



Coarsest level – Slit boundary

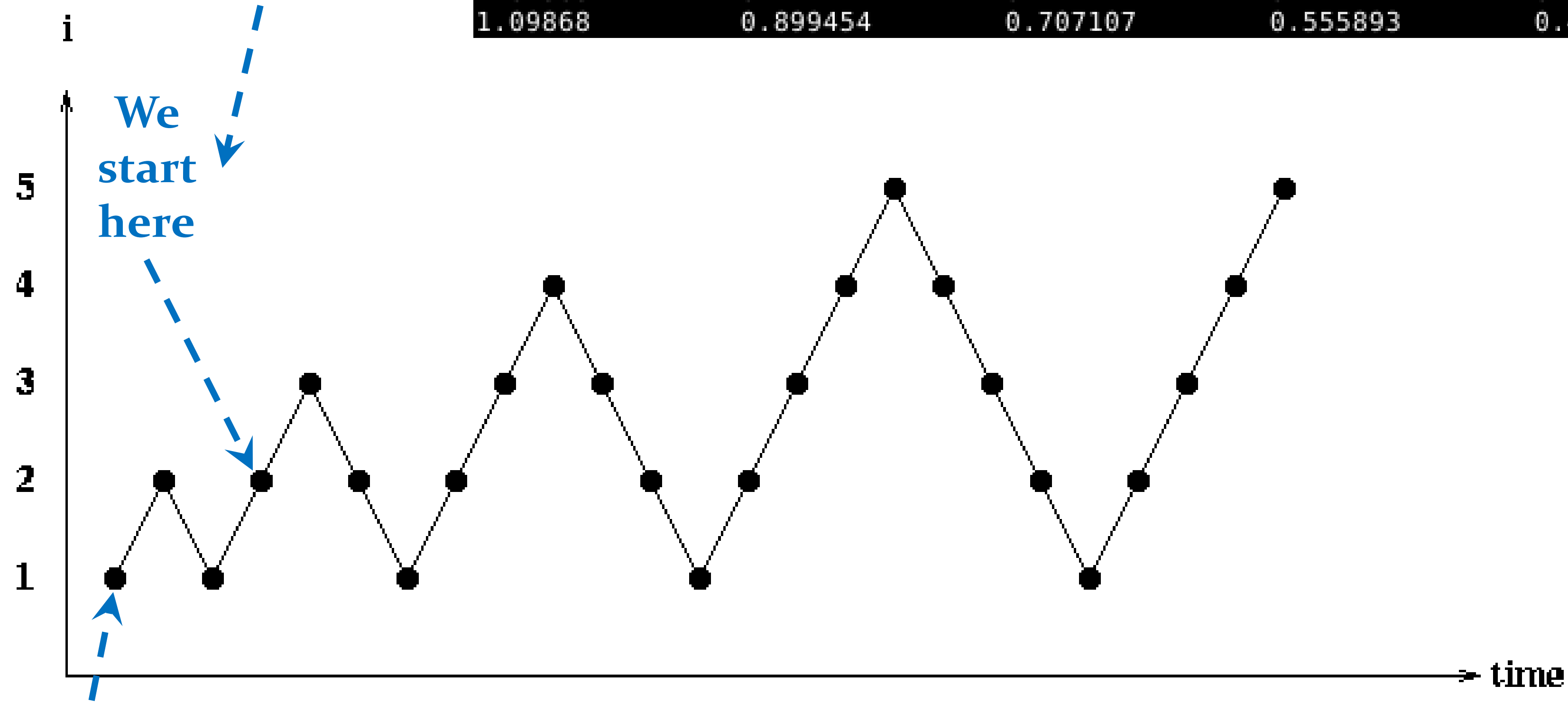
V-cycles

Full Multigrid cycle

Set BCs

```
u after applying FMG BCs
Grid Entries:
1.09868      0.899454     0.707107     0.555893     0.45509
1.02909      0            0            0            0.242934
1            0            0            0            0
1.02909      0            0            0            0.242934
1.09868      0.899454     0.707107     0.555893     0.45509
```

We start here



Coarsest level – Slit boundary

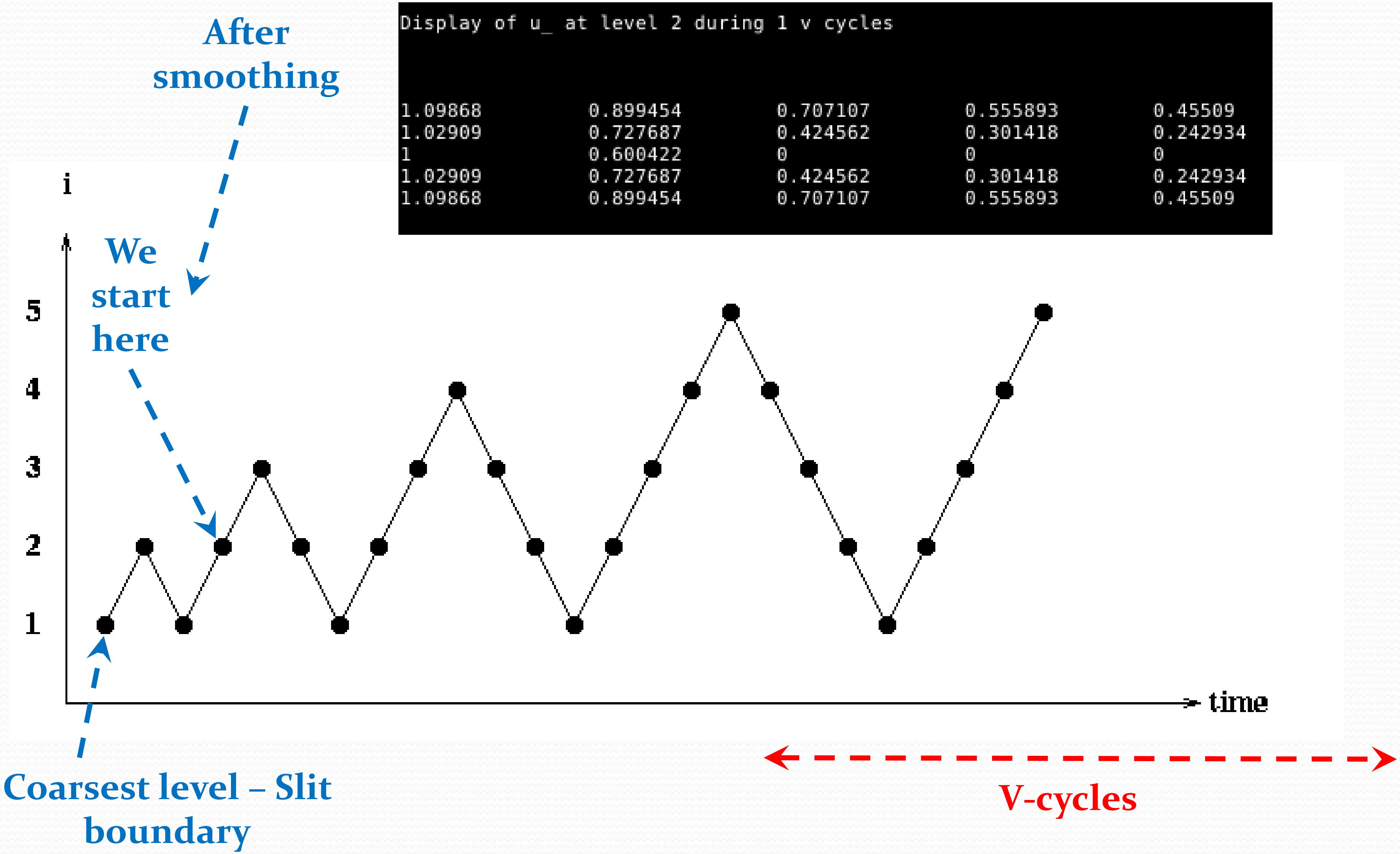
V-cycles

Full Multigrid cycle

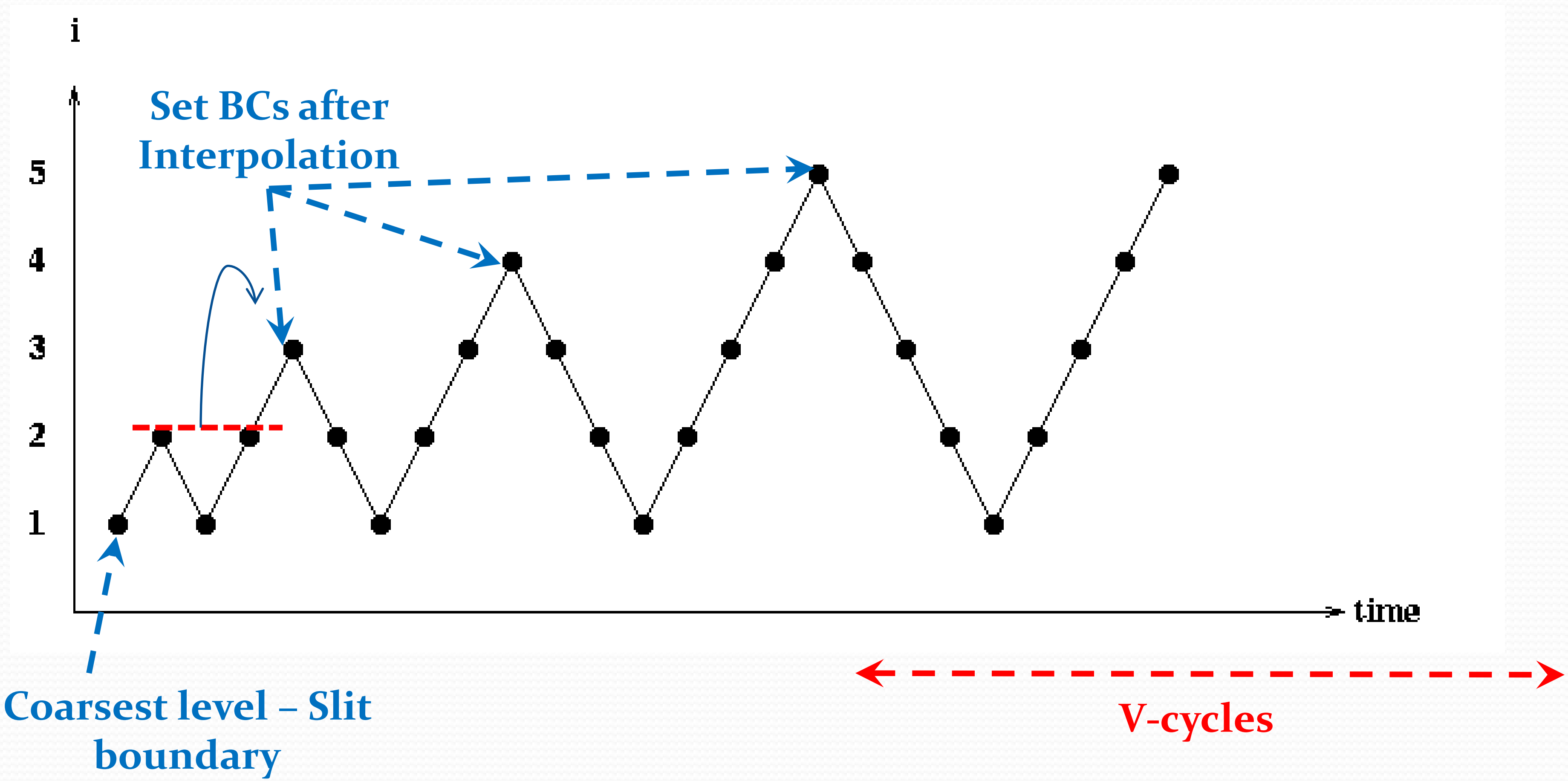
```
Display of u_ at level 2 during 1 v cycles



1.09868         0.899454        0.707107        0.555893        0.45509
1.02909         0.727687        0.424562        0.301418        0.242934
1               0.600422        0               0               0
1.02909         0.727687        0.424562        0.301418        0.242934
1.09868         0.899454        0.707107        0.555893        0.45509
```

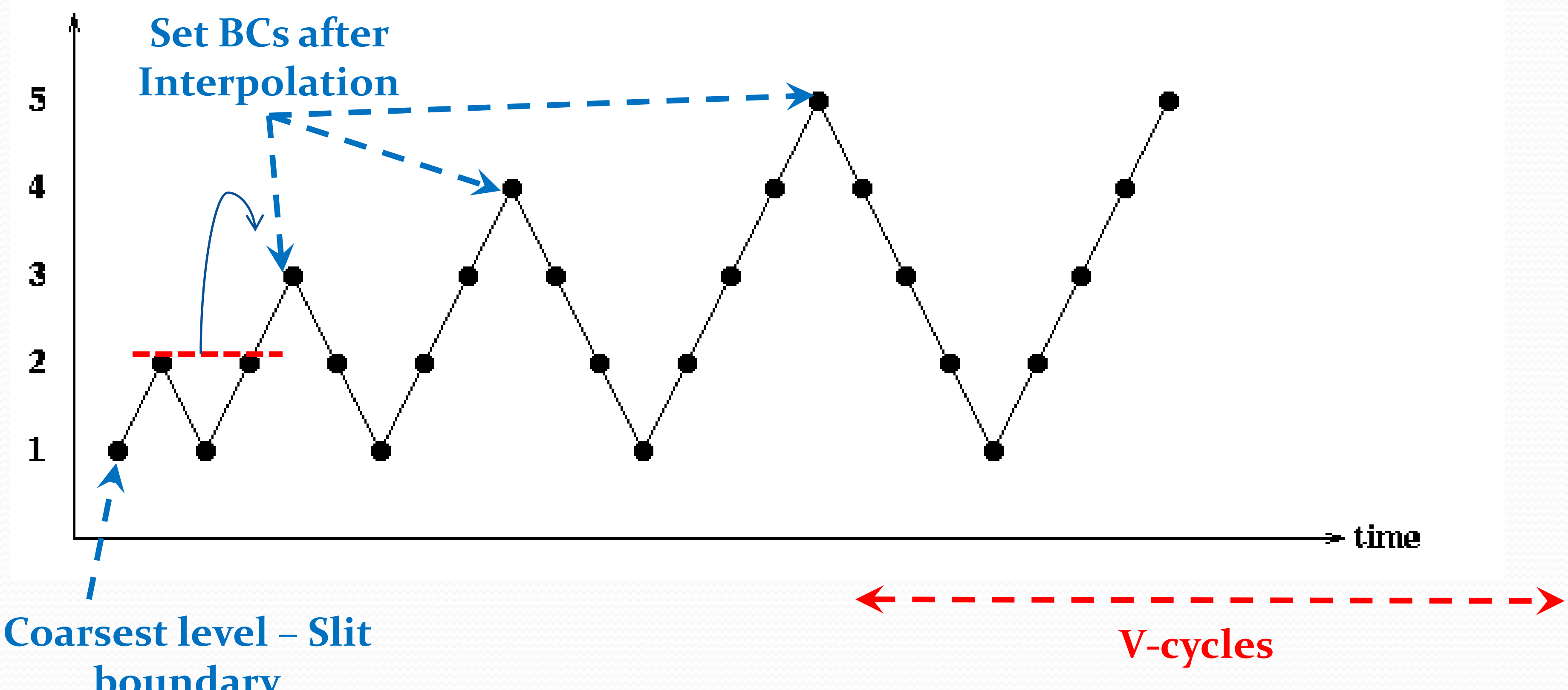


Full Multigrid cycle



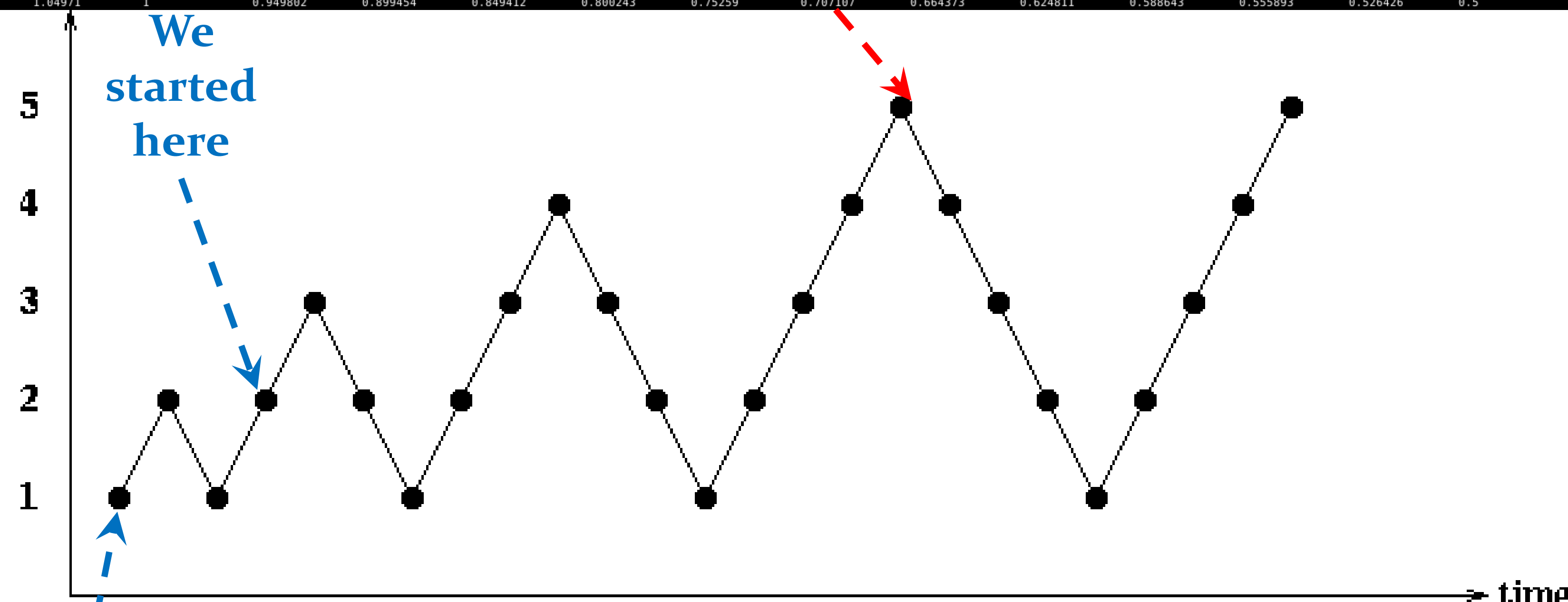
Full Multigrid cycle

```
levelFMG in FMGsetBCs() is :3
u after applying FMG BCs
Grid Entries:
1.09868    1          0.899454   0.800243   0.707107   0.624811   0.555893   0.5        0.45509
1.06066    0.938728   0.81357    0.689702   0.565835   0.497245   0.428655   0.388834   0.353553
1.02909    0.878386   0.727687   0.576125   0.424562   0.36299    0.301418   0.272176   0.242934
1.00766    0.839299   0.664054   0.438168   0.212281   0.181495   0.150709   0.136088   0.124049
1          0.800211   0.600422   0.300211   0          0          0          0          0
1.00766    0.839299   0.664054   0.438168   0.212281   0.181495   0.150709   0.136088   0.124049
1.02909    0.878386   0.727687   0.576125   0.424562   0.36299    0.301418   0.272176   0.242934
1.06066    0.938728   0.81357    0.689702   0.565835   0.497245   0.428655   0.388834   0.353553
1.09868    1          0.899454   0.800243   0.707107   0.624811   0.555893   0.5        0.45509
```



Full Multigrid cycle

1.09868	1.04971	1	0.949802	0.899454	0.849412	0.800243	0.75259	0.707107	0.664373	0.624811	0.588643	0.555893	0.526426	0.5	0.476324	0.45509
1.07907	1.02723	0.974317	0.92057	0.866365	0.812261	0.759011	0.707529	0.65878	0.613589	0.572468	0.535572	0.50277	0.473755	0.448133	0.425492	0.405443
1.06066	1.00582	0.949471	0.891795	0.833176	0.774257	0.716011	0.659737	0.606894	0.558737	0.515902	0.478406	0.445862	0.417692	0.393284	0.372069	0.353553
1.04385	0.985937	0.925947	0.863964	0.800288	0.73558	0.67104	0.608511	0.550323	0.498562	0.453995	0.416291	0.384578	0.357866	0.335242	0.315946	0.299372
1.02909	0.968125	0.904416	0.837827	0.76843	0.696736	0.624056	0.552946	0.487326	0.431192	0.385228	0.348182	0.318294	0.293951	0.273873	0.2571	0.242934
1.01686	0.953062	0.885767	0.814496	0.738869	0.658879	0.575503	0.49189	0.414844	0.353651	0.307541	0.272917	0.246465	0.225772	0.209198	0.195649	0.184392
1.00766	0.941498	0.871094	0.79552	0.713671	0.624407	0.527188	0.424266	0.326508	0.261029	0.218368	0.189478	0.168877	0.153474	0.141499	0.131906	0.124049
1.00194	0.934172	0.861591	0.782819	0.69589	0.597888	0.484576	0.351478	0.205893	0.145587	0.115426	0.0977493	0.0860932	0.0777461	0.0714176	0.0664255	0.0623788
1	0.931656	0.858278	0.778275	0.689183	0.586678	0.461752	0.291177	0	0	0	0	0	0	0	0	0
1.00194	0.934172	0.861591	0.782819	0.69589	0.597888	0.484576	0.351478	0.205893	0.145587	0.115426	0.0977493	0.0860932	0.0777461	0.0714176	0.0664255	0.0623788
1.00766	0.941498	0.871094	0.79552	0.713671	0.624407	0.527188	0.424266	0.326508	0.261029	0.218368	0.189478	0.168877	0.153474	0.141499	0.131906	0.124049
1.01686	0.953062	0.885767	0.814496	0.738869	0.658879	0.575503	0.49189	0.414844	0.353651	0.307541	0.272917	0.246465	0.225772	0.209198	0.195649	0.184392
1.02909	0.968125	0.904416	0.837827	0.76843	0.696736	0.624056	0.552946	0.487326	0.431192	0.385228	0.348182	0.318294	0.293951	0.273873	0.2571	0.242934
1.04385	0.985937	0.925947	0.863964	0.800288	0.73558	0.67104	0.608511	0.550323	0.498562	0.453995	0.416291	0.384578	0.357866	0.335242	0.315946	0.299372
1.06066	1.00582	0.949471	0.891795	0.833176	0.774257	0.716011	0.659737	0.606894	0.558737	0.515902	0.478406	0.445862	0.417692	0.393284	0.372069	0.353553
1.07907	1.02723	0.974317	0.92057	0.866365	0.812261	0.759011	0.707529	0.65878	0.613589	0.572468	0.535572	0.50277	0.473755	0.448133	0.425492	0.405443
1.09868	1.04971	1	0.949802	0.899454	0.849412	0.800243	0.75259	0.707107	0.664373	0.624811	0.588643	0.555893	0.526426	0.5	0.476324	0.45509

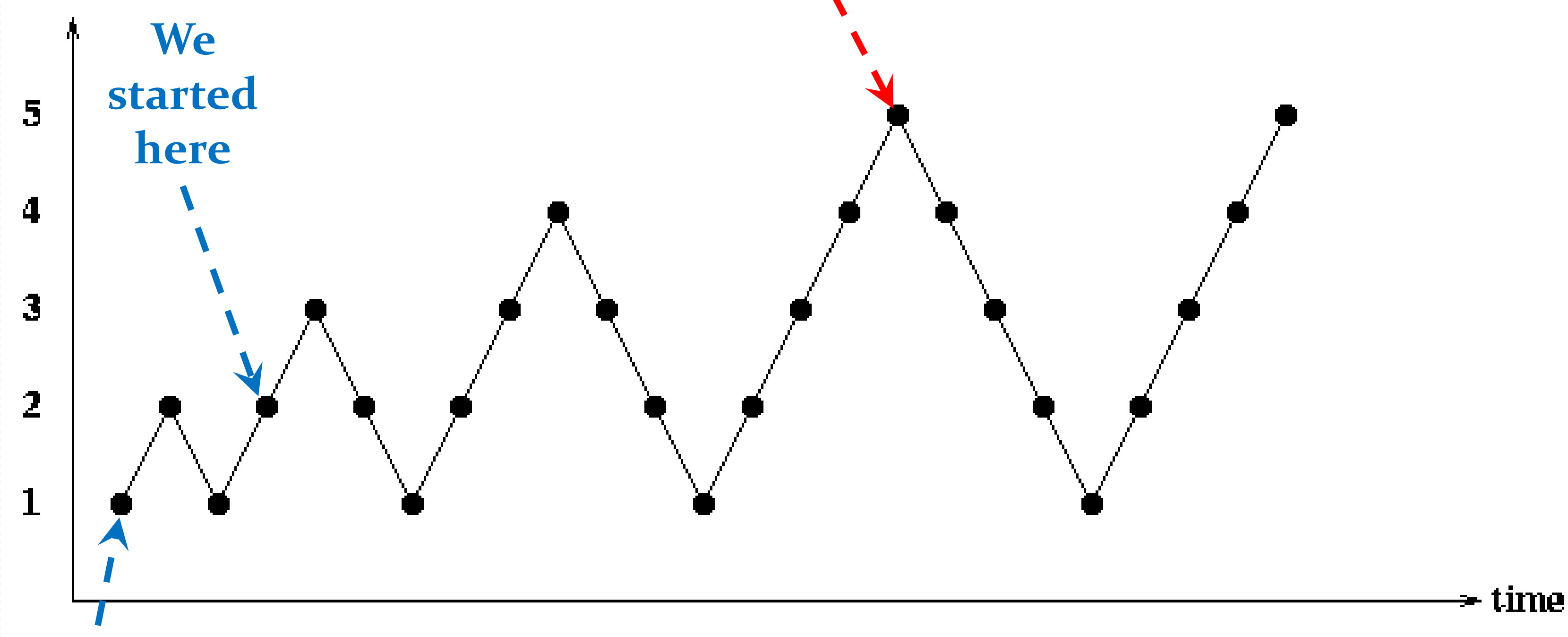


Coarsest level – Slit boundary

V-cycles

If (!Full Multigrid cycle)

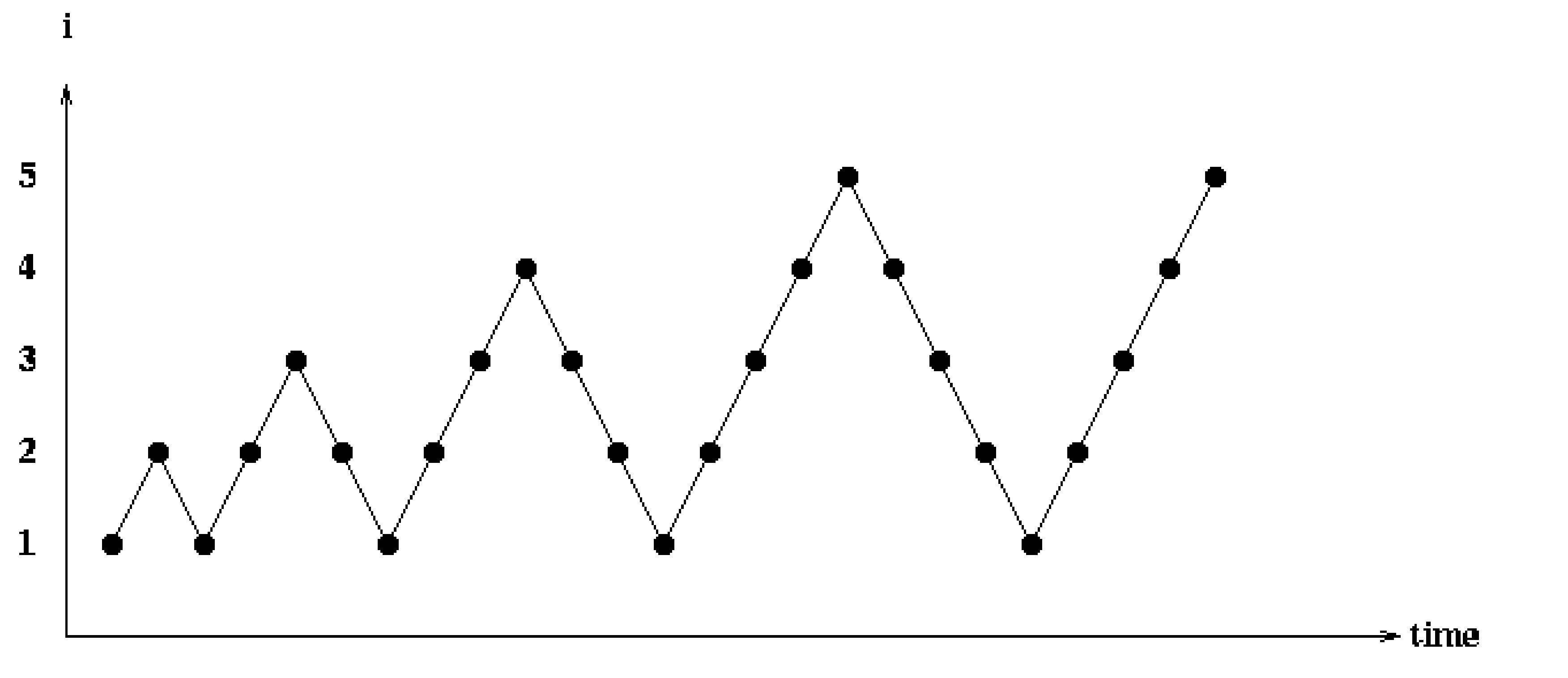
1.09868	1.04971	1	0.949802	0.899454	0.849412	0.800243	0.75259	0.707107	0.664373	0.624811	0.588643	0.555893	0.526426	0.5	0.476324	0.45509
1.07907	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.405443
1.06066	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.353553
1.04385	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.299372
1.02909	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.242934
1.01686	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.184392
1.00766	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.124049
1.00194	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.0623788
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1.00194	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.0623788
1.00766	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.124049
1.01686	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.184392
1.02909	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.242934
1.04385	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.299372
1.06066	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.353553
1.07907	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.405443
1.09868	1.04971	1	0.949802	0.899454	0.849412	0.800243	0.75259	0.707107	0.664373	0.624811	0.588643	0.555893	0.526426	0.5	0.476324	0.45509



Coarsest level – Slit boundary

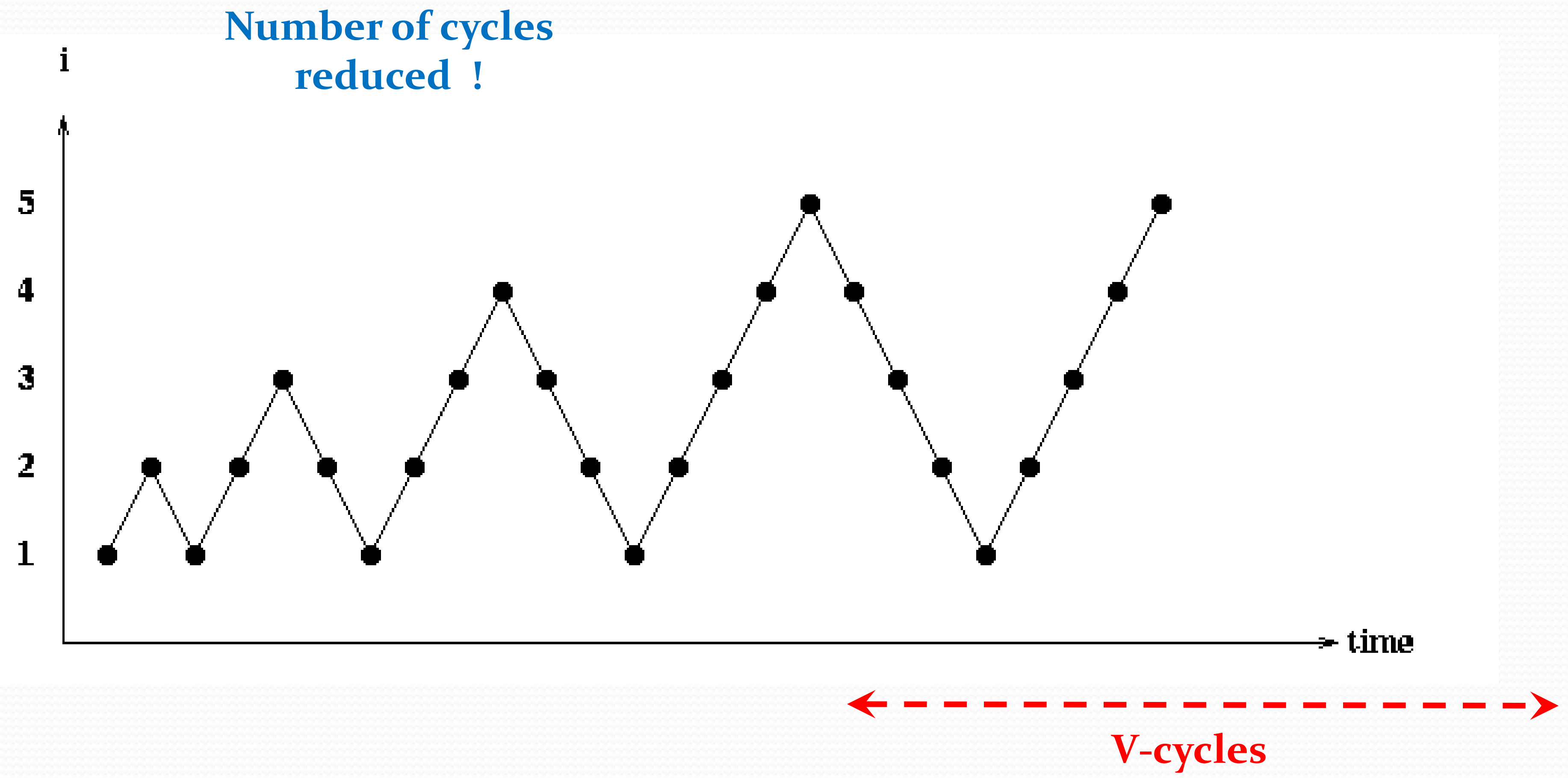
V-cycles

Convergence ?



V-cycles

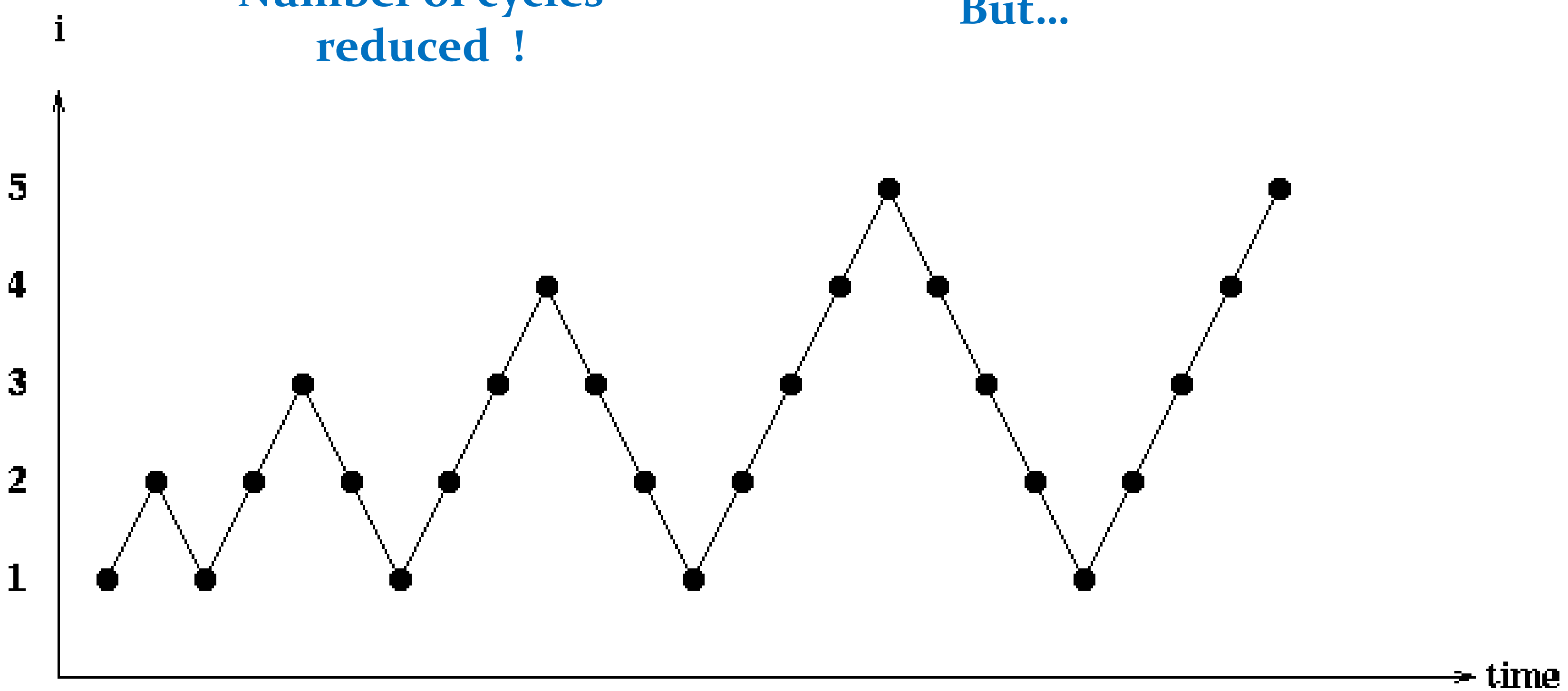
Convergence ?



Convergence ?

Number of cycles
reduced !

But…



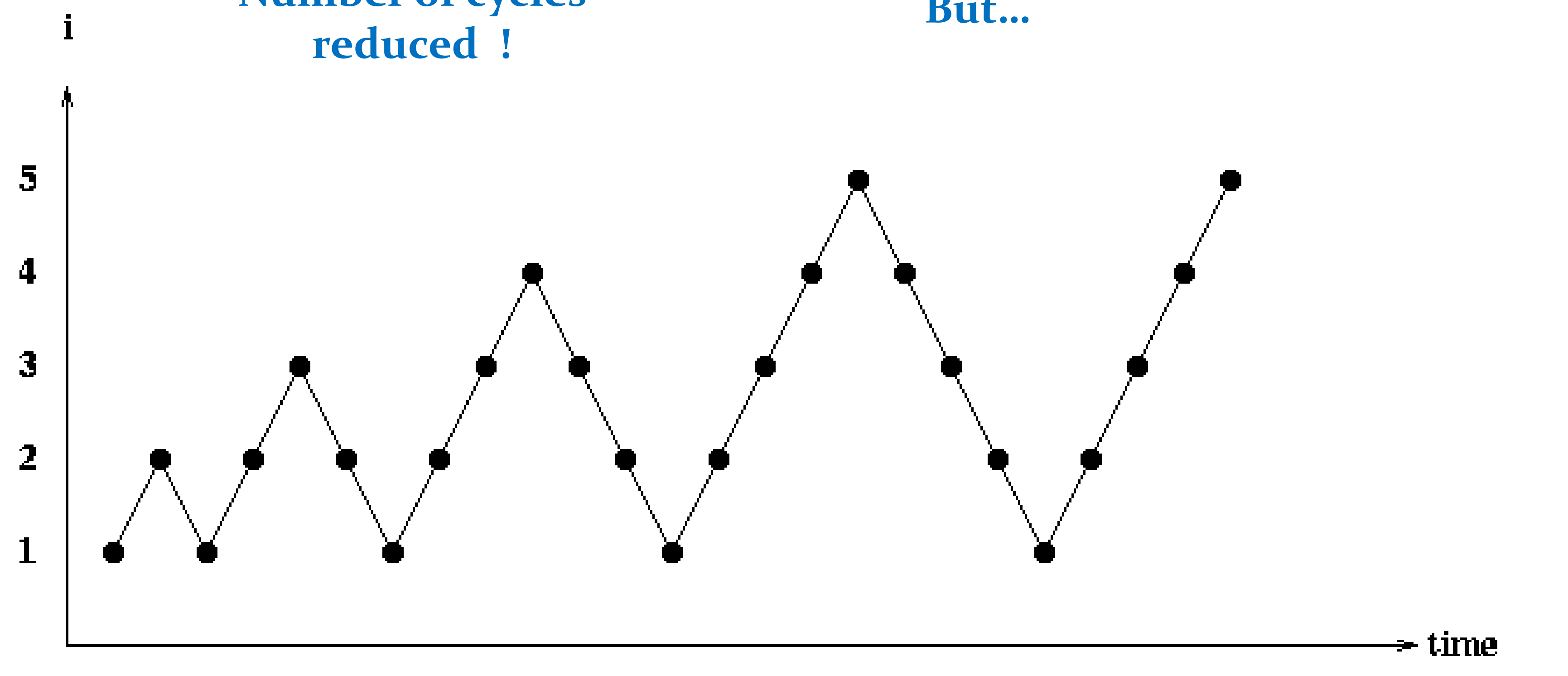
V-cycles

Convergence ?

BleedBlue	0,459698	1,94895388093e-05	3. Juni 2016	i10hpc2	Serial

Number of cycles reduced !

But…



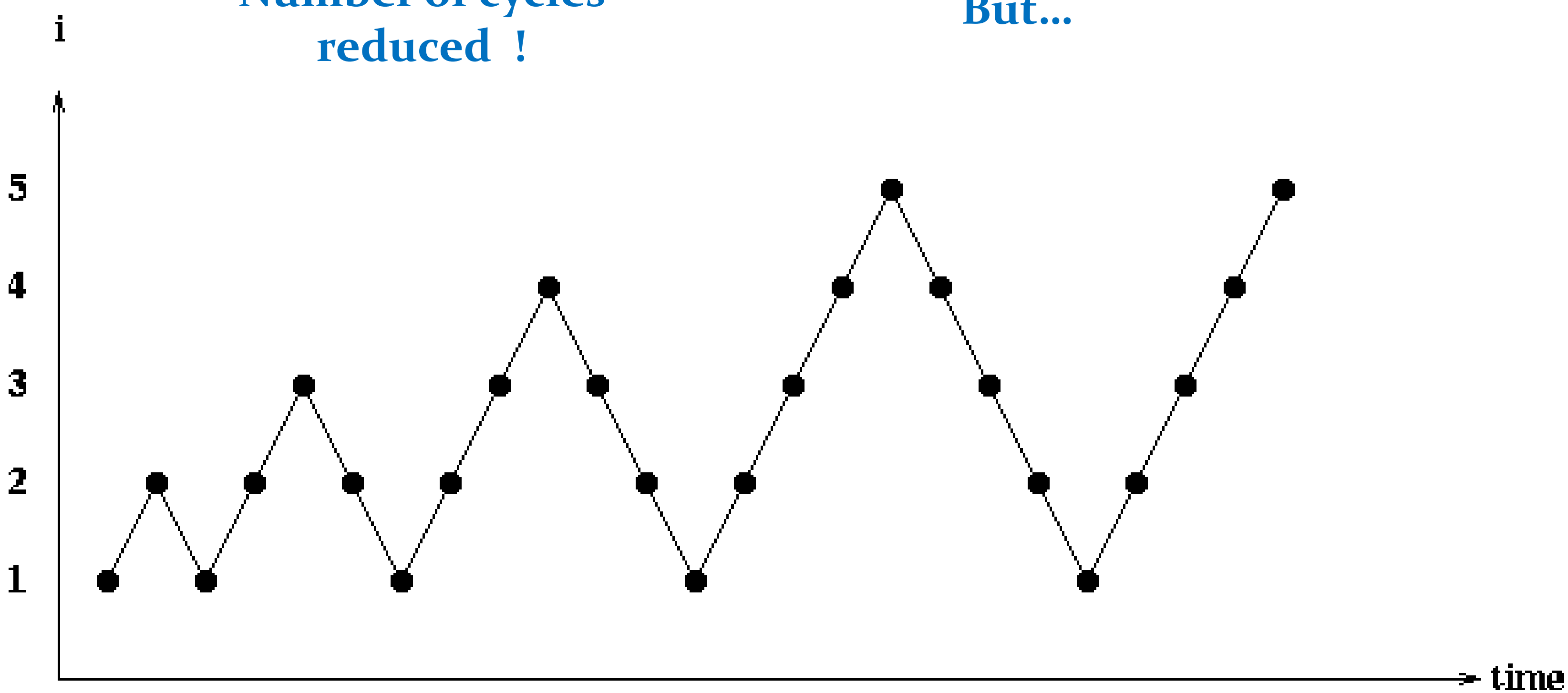
V-cycles

Convergence ?

BleedBlue	0,459698	1,94895388093e-05	3. Juni 2016	i10hpc2	Serial

Number of cycles reduced !

But…



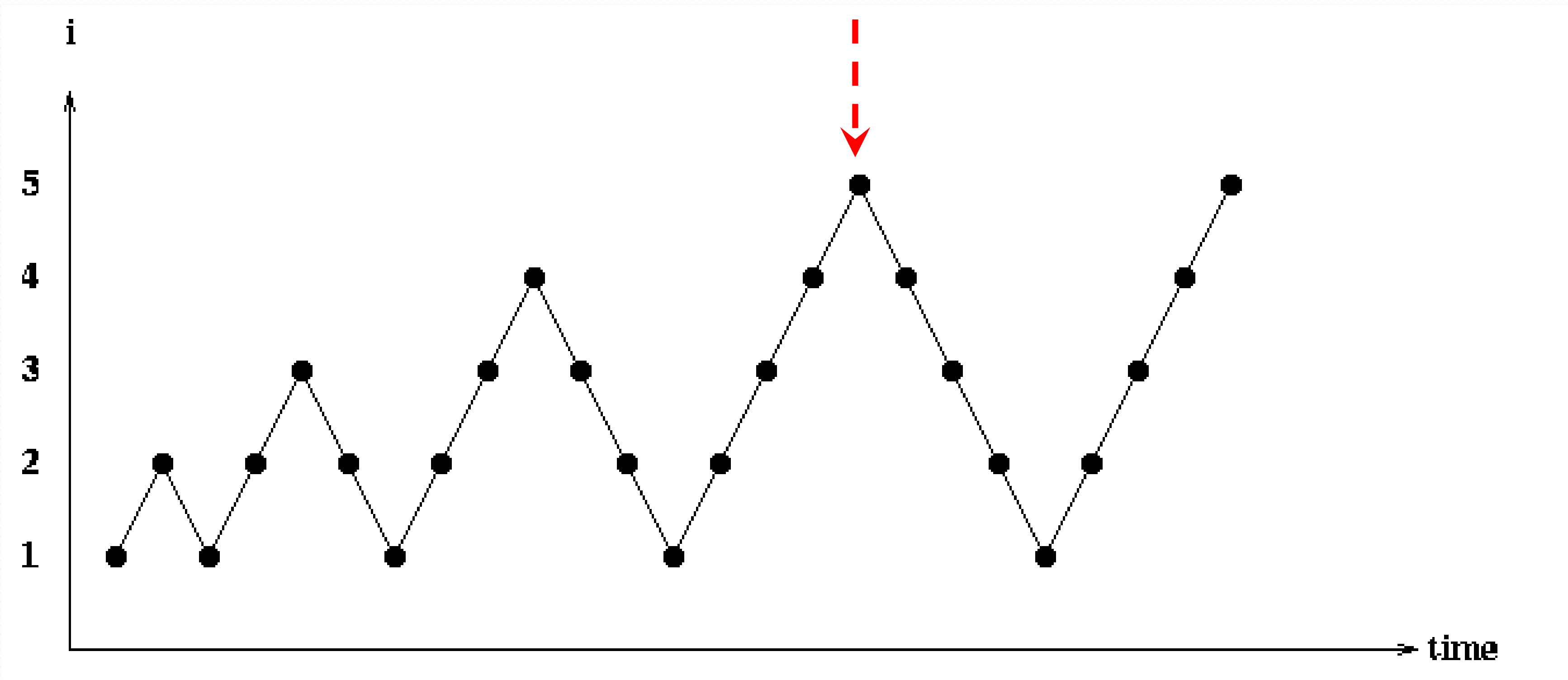
V-cycles

Error plot

BleedBlue	0,459698	1,94895388093e-05	3. Juni 2016	i10hpc2	Serial

Challenges in FMG

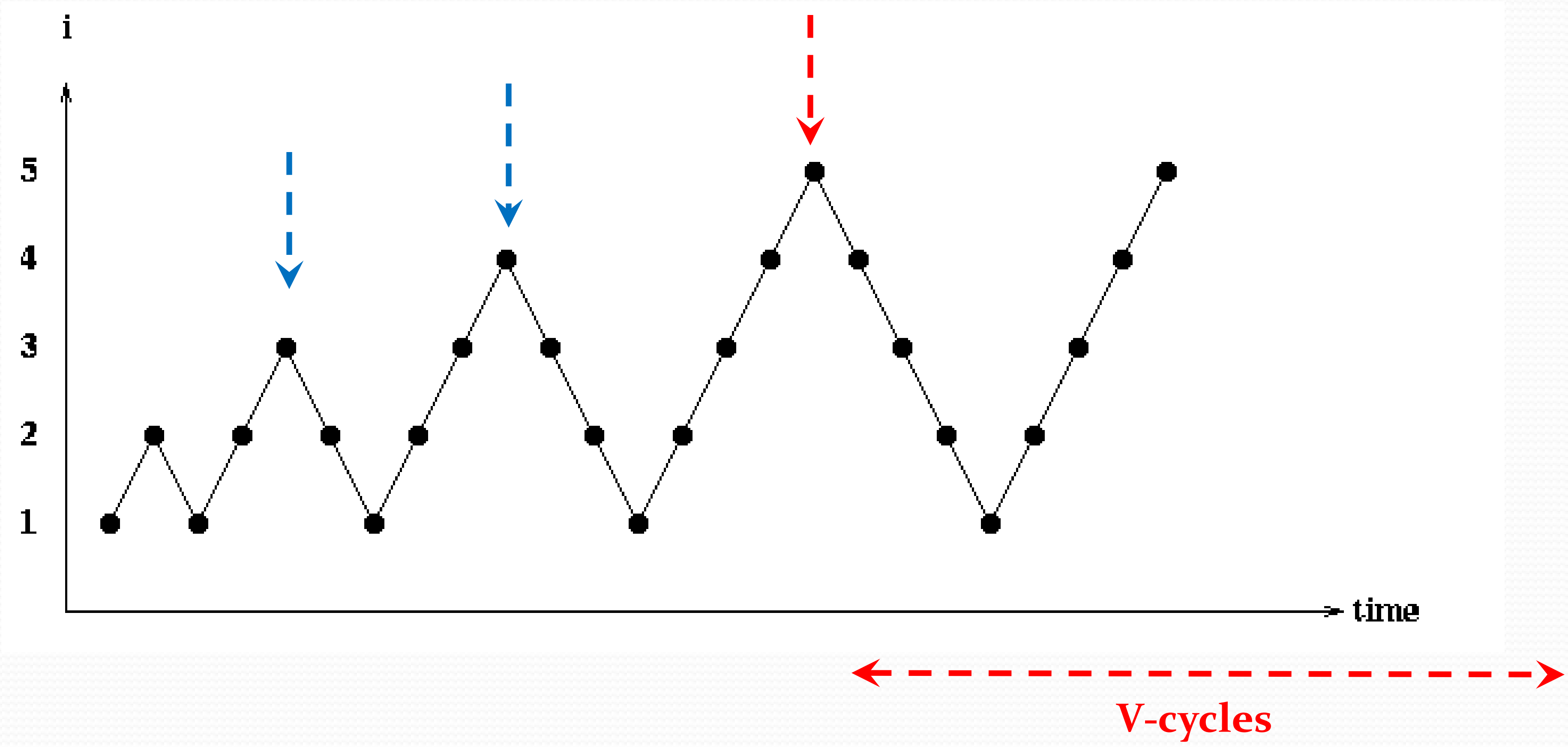
Reset array values of coarser levels to zero



V-cycles

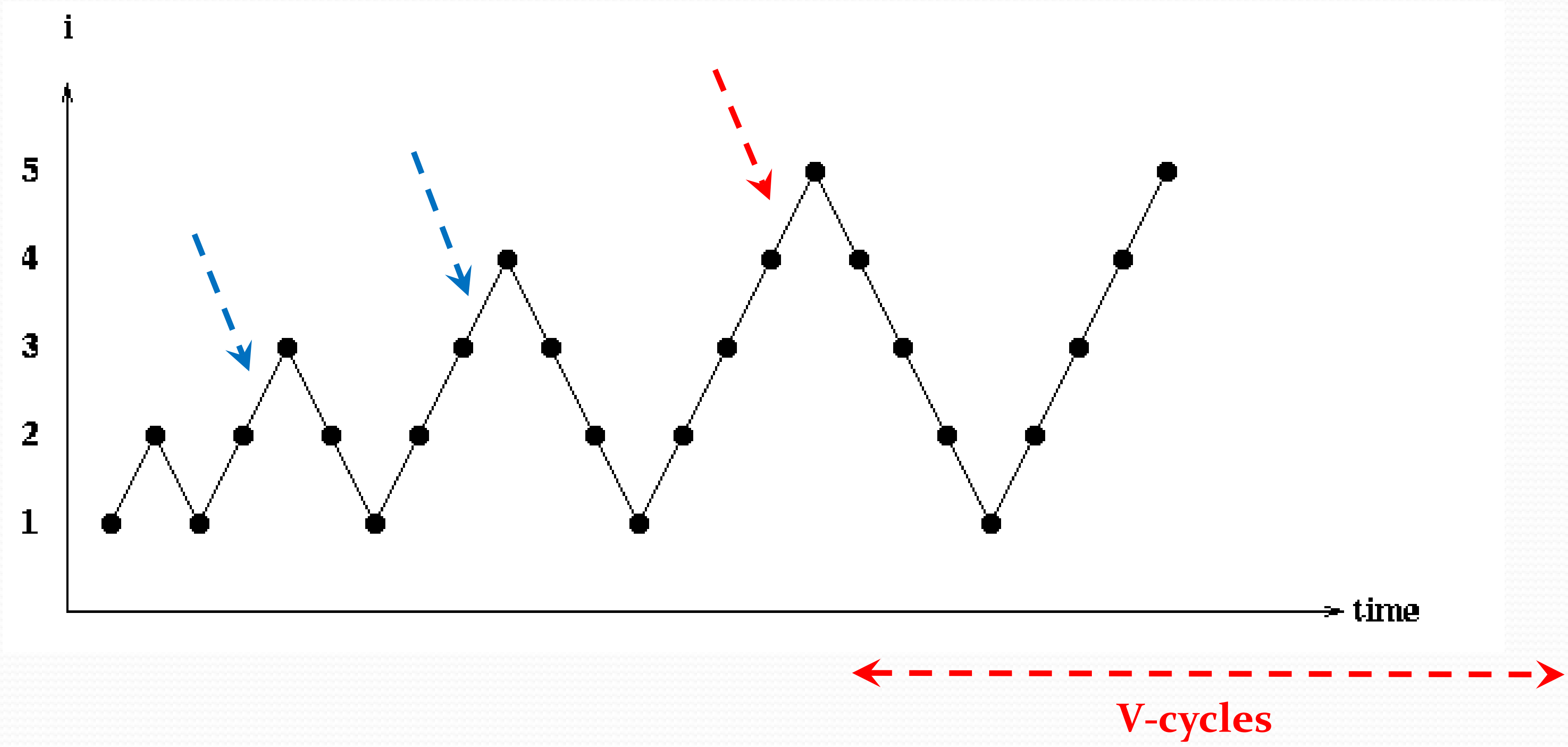
Challenges in FMG

Reset array values of
coarser levels to zero



Challenges in FMG

**Changes in the
Interpolation method**



Parallelization

- OpenMP

- #pragma omp parallel for schedule(static)
  - MultiGridSolver::applyRBGS_Iter() Red Black Smoother
  - MultiGridSolver::applyRestriction() Restriction Operator
  - MultiGridSolver::applyInterpolation() Interpolation Operator
  - MultiGridSolver::computeResidual() Residual Computation

Parallelization

- OpenMP - Only V - Cycles

- #CPUs (OpenMP option on Verifier) Runtime of MG Solver (seconds)

#CPUs	Runtime
1	**9.48118**
2	**4.53418**
4	**2.21559**
8	**1.10856**
16	**1.0314**
32	**0.68125**

Serial option on cluster **0.65112**

Parallelization

- OpenMP – V - Cycles with **FMG**

- #CPUs (OpenMP option on Verifier) Runtime of MG Solver (seconds)

#CPUs (OpenMP option on Verifier)	Runtime of MG Solver (seconds)
1	**5.46958**
2	**3.84408**
4	**1.9959**
8	**0.90064**
16	**1.10745**
32	**0.542416**

Serial option on cluster **0.459698**

Vectorization

- Non- Vectorized code

  - ./mgsolve 12 1 for non Vectorized Code

    Your Alias: BleedBlue
    Wall clock time of applyRBGS_Iter() is **62.596** ms
    Wall clock time of compute Residual()) is **76.07** ms
    Wall clock time of Restriction() is **31.649** ms
    Wall clock time of applyRBGS_Iter() is **60.905** ms
    Wall clock time of MG execution: **369.419** ms

Vectorization

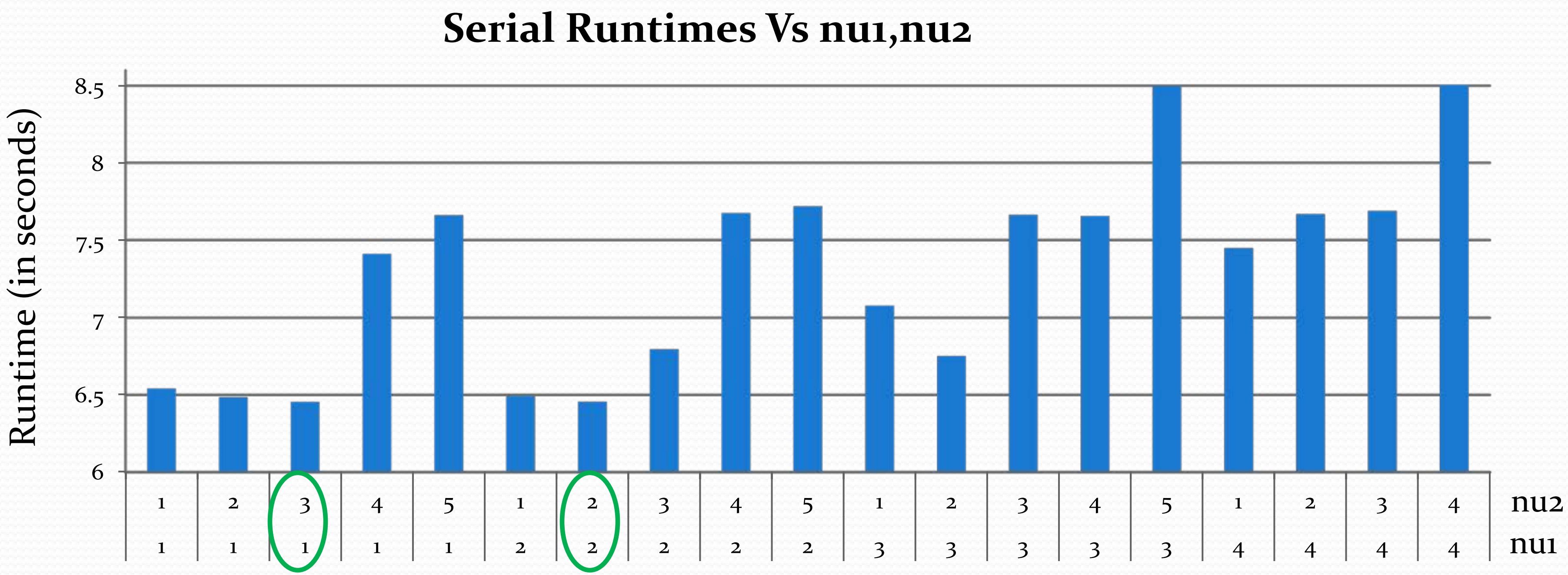
- Vectorized code
  - ./mgsolve 12 1 for non Vectorized Code

    Your Alias: BleedBlue
    Wall clock time of applyRBGS_Iter() is **157.469** seconds
    Wall clock time of compute Residual() is **246.207** seconds
    Wall clock time of Restriction() in **91.559** seconds
    Wall clock time of applyRBGS_Iter() in **157.43** seconds
    Wall clock time of MGMSolve() in **1002.68** seconds
    Wall clock time of MG execution: **1002.72** ms

Vectorization

Suggestions ?

Comparison of Smoothening Parameters



- Variation of serial runtimes on the LSS cluster with the number of pre-smoothing(nu 1) and post smoothing steps (nu 2) has been studied.
- This study was done using the RBGS Solver for intermediate steps.
- The combinations – (1,3) and (2,2) show the least runtimes and took 11 v-cycles each for convergence.

Comparison of Smoothening Parameters (contd.)

nu1	nu2	V-cycles for Convergence	Error	Cluster Serial Runtime (in seconds)
1	1	16	9.17732E-05	6.5414
1	2	13	9.17619E-05	6.48613
1	3	11	9.17927E-05	6.45551
1	4	11	9.17443E-05	7.40991
1	5	10	9.17568E-05	7.66071
2	1	13	9.17693E-05	6.49229
2	2	11	9.17789E-05	6.45584
2	3	10	9.17806E-05	6.79593
2	4	10	9.17454E-05	7.67483
2	5	9	9.17715E-05	7.71864
3	1	12	9.17514E-05	7.0785
3	2	10	9.17885E-05	6.75157
3	3	10	9.17448E-05	7.66405
3	4	9	9.17670E-05	7.65531
3	5	9	9.17462E-05	8.49673
4	1	11	9.17549E-05	7.44863
4	2	10	9.17507E-05	7.66858
4	3	9	9.17707E-05	7.68896
4	4	9	9.17463E-05	8.50611

- A comparative study of convergence rates for different iterative solvers has been done for these two combinations.

Comparison of Different Solvers

V-cycles to convergence, Normalized Runtimes for nu1 =1, nu2 =3				
Solver	**V-cycles for Convergence**	**Error**	**Serial Runtime (in ms)**	**Serial Runtime (Normalized)**
Jacobi	16	9.17733E-05	14201.1	1.0000
Damped Jacobi, w = 1/3	37	9.17984E-05	35672.3	2.5119
Damped Jacobi, w = 1/2	26	9.17884E-05	25245.8	1.7777
Damped Jacobi, w = 2/3	21	9.17702E-05	20299.2	1.4294
Natural Gauss Seidel (NGS)	12	9.17586E-05	10829	0.7625
SOR on NGS, w = 4/3	9	9.17465E-05	9058.38	0.6379
SOR on NGS, w = 5/3	7	9.17399E-05	7003.42	0.4932
RBGS	11	9.17904E-05	9862.52	0.6945
SOR on RBGS, w = 4/3	8	9.17398E-05	8022.64	0.5649
SOR on RBGS, w = 5/3	7	9.13771E-05	7017.09	0.4941
SOR on RBGS, w = 11/6	12	9.1712E-05	12015.7	0.8461

- SOR on NGS works even better than SOR on RBGS with w = 5/3, but since it can't be parallelized, the SOR on RBGS option is better.

Comparison of Different Solvers (contd.)

V-cycles to convergence, Normalized Runtimes for nu1 =2, nu2 =2				
Solver	**V-cycles for Convergence**	**Error**	**Serial Runtime (in ms)**	**Serial Runtime (Normalized)**
Jacobi	16	9.17713E-05	14243.5	1.0000
Damped Jacobi, w = 1/3	37	9.17936E-05	35729.2	2.5085
Damped Jacobi, w = 1/2	26	9.17799E-05	25358	1.7803
Damped Jacobi, w = 2/3	21	9.17621E-05	20315.2	1.4263
Nautral Gauss Seidel	12	9.17474E-05	10854.5	0.7621
SOR on NGS, w = 4/3	9	9.17455E-05	9076.22	0.6372
SOR on NGS, w = 5/3	7	9.17352E-05	7018.52	0.4928
RBGS	11	9.17767E-05	9885.48	0.6940
SOR on RBGS with w = 4/3	7	9.17847E-05	7026.47	0.4933
SOR on RBGS with w = 5/3	9	9.09422E-05	9024.19	0.6336

- For this case, faster convergence is achieved for SOR on RBGS with w=4/3.
- Damped Jacobi Solver shows worse performance than Jacobi Iteration .
- Thus, nu 1=2,nu 2=2 for SOR on RBGS with w=4/3 seems to be the optimum choice for implementation.

THANK YOU