

vl·e



virtual laboratory for e-science

# From gridified scripts to workflows: the FSL *Feat* case

Tristan Glatard and Sílvia D. Olabbarriaga

Academic Medical Center – Informatics Institute  
University of Amsterdam

MICCAI-G workshop – September 6<sup>th</sup> 2008



# Workflows in neuroimaging

- Coming up in the community
  - See e.g. [Rex *et al* 03, Porro *et al* 06, Fissel 08, Soleman *et al* 08, Krefting *et al* 08, Pernod *et al* 08]
- Transparency of analysis methods
  - Eases application tweaking
  - Improves reusability & maintenance (components)
  - Improves error detection
- Facilitated access to grids
  - Transparent parallelization
  - Performance improvement (↓ CPU time, ↓ results size)





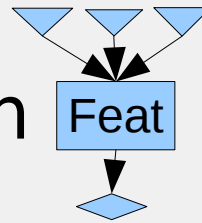
# Workflow drawbacks

- Performance issues
  - ↑ number of jobs (↑ grid load, ↑ fault probability)
  - ↑ data transfers
  - ↑ sensitivity to latency
- Usability issues
  - Tiresome description of the application
  - Management of distributed results

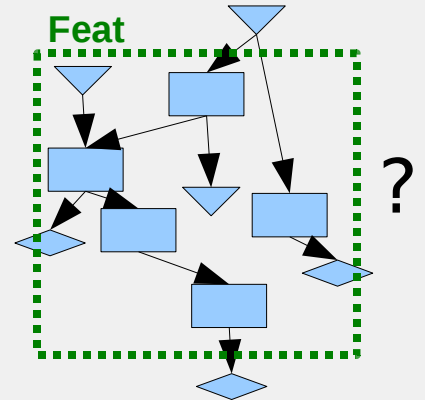


# Outline

Is it worth moving from



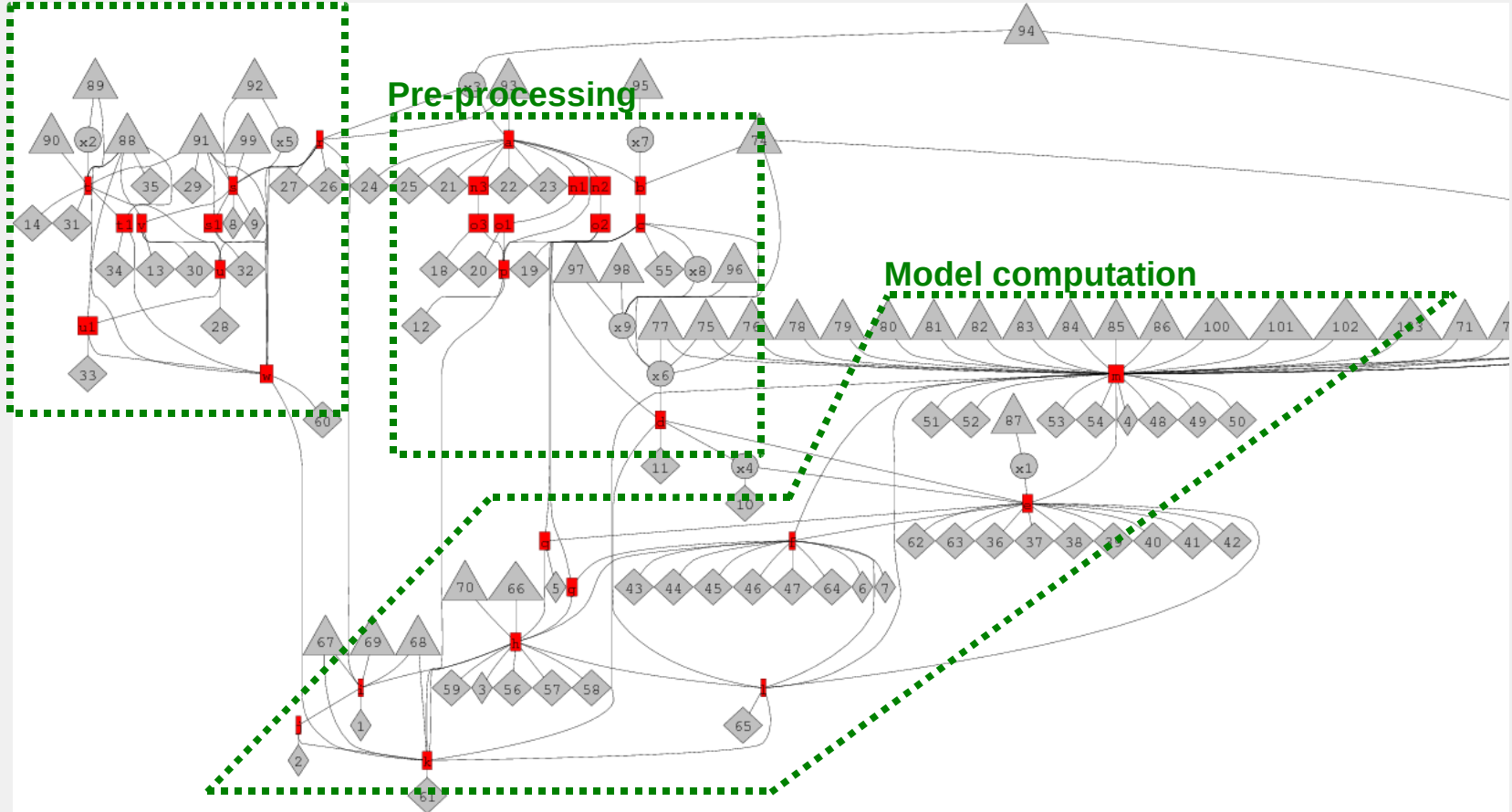
to



- Introduction
- Workflow implementation description
- Performance comparison
- Output organization

# Feat FSL workflow

## Normalization



- Largest workflow on **my experiment** <sup>beta</sup> (in June 2008)
- To be iterated hundreds to thousands of times
- Used Scufi language with dot-product from [Montagnat *et al*'06]
- Expected parallelism exploitation



# Implementation evaluation

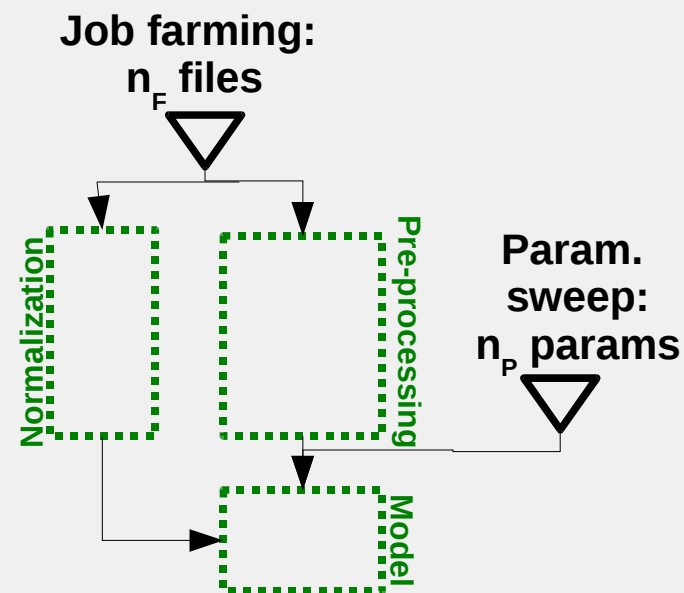
- ✓ Reproduced use-case of [Soleman *et al* 08]
  - Assessed on limited dataset
  - Executed on vlemmed EGEE VO using MOTEUR
- ✗ Not implemented *Feat* options
  - B0 unwarping, contrast masking, denoising, perfusion subtraction
- ✗ Dynamic patterns hardly manageable
  - e.g., fixed number of EVs and contrasts
- ✗ May not generalize to other use-cases
  - Assumed, e.g., 1 anatomical scan *per* EPI scan



# Performance study

- Use-cases

- Job farming ( $n_F$  files)
- Sweep on model parameter ( $n_p$  parameters)

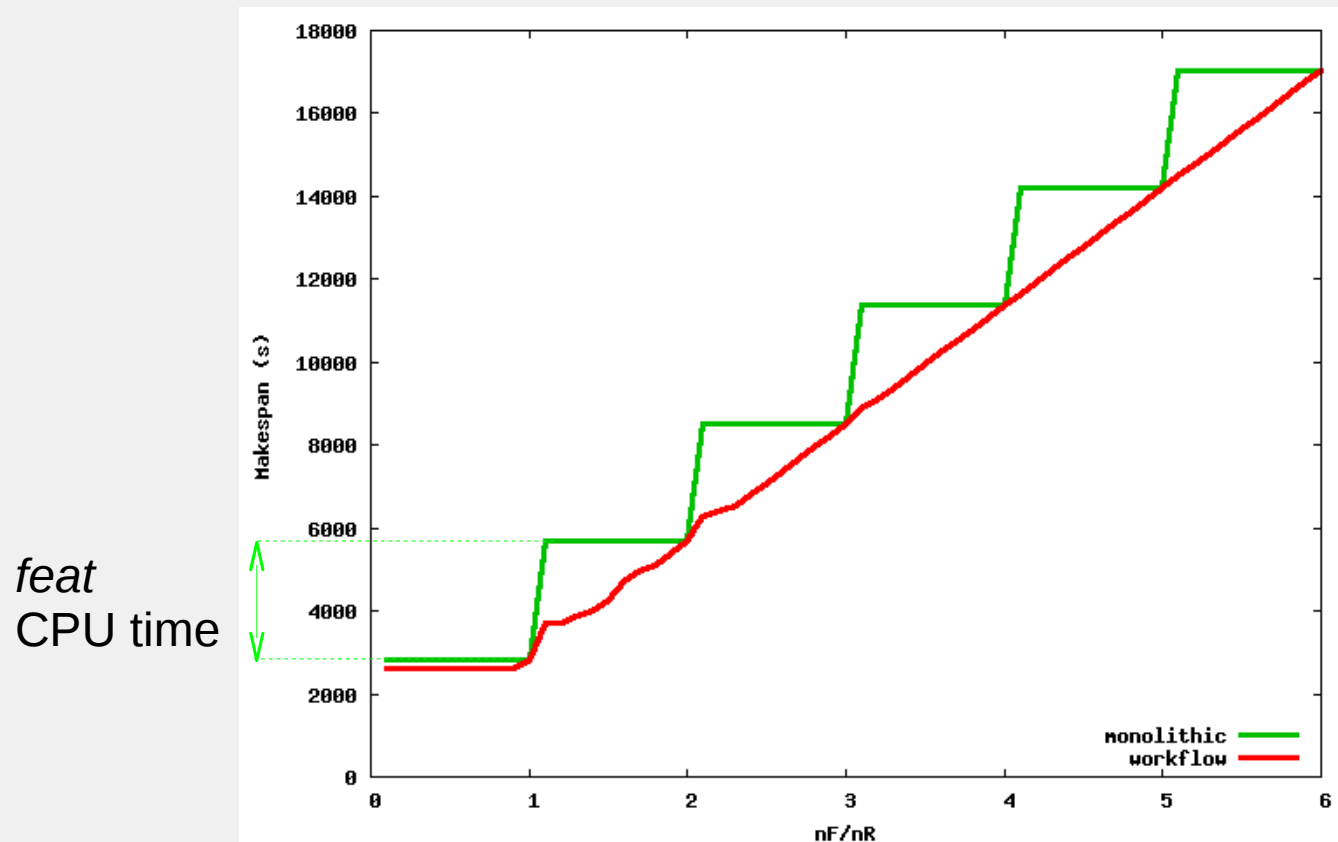


- Simulation of workflow scheduling

- List-scheduling algorithm ( $n_R = 10$  resources)
- Data transfers measured on vlemmed VO
- CPU time measured on local PC
- With/without latency: time to access free resource

# Results: job farming

- No data transfers - no latency

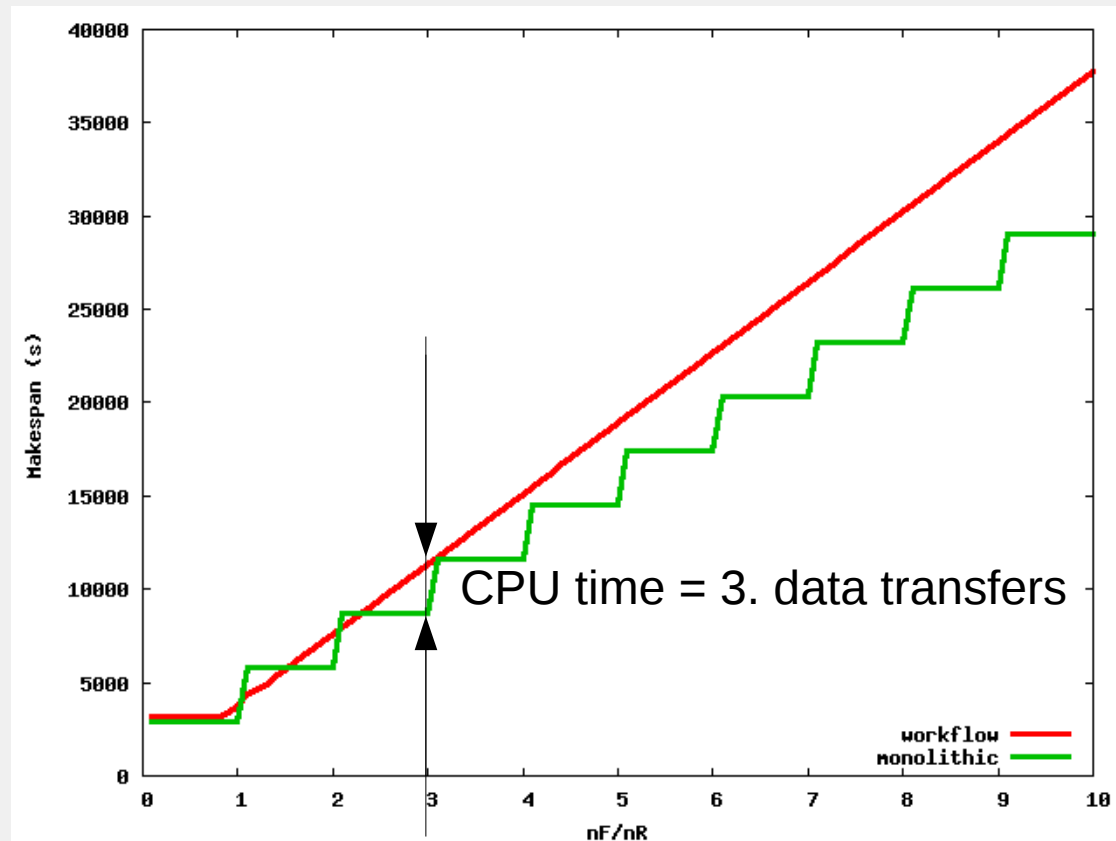


- Workflow outperforms monolithic
- Reaches linear speed-up



# Results: job farming (#2)

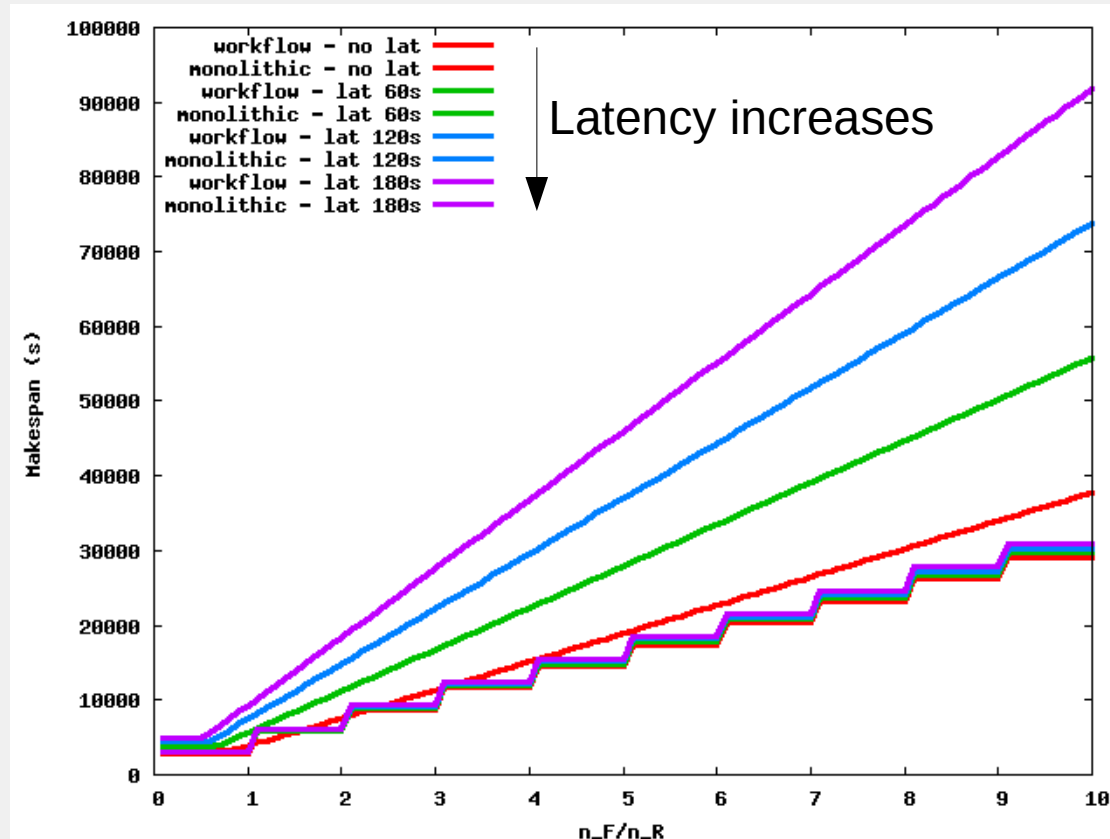
- With data transfers - no latency



- Workflow similar to monolithic up to  $n_F = 3.n_R$

# Results: job farming (#3)

- With data transfers and latency

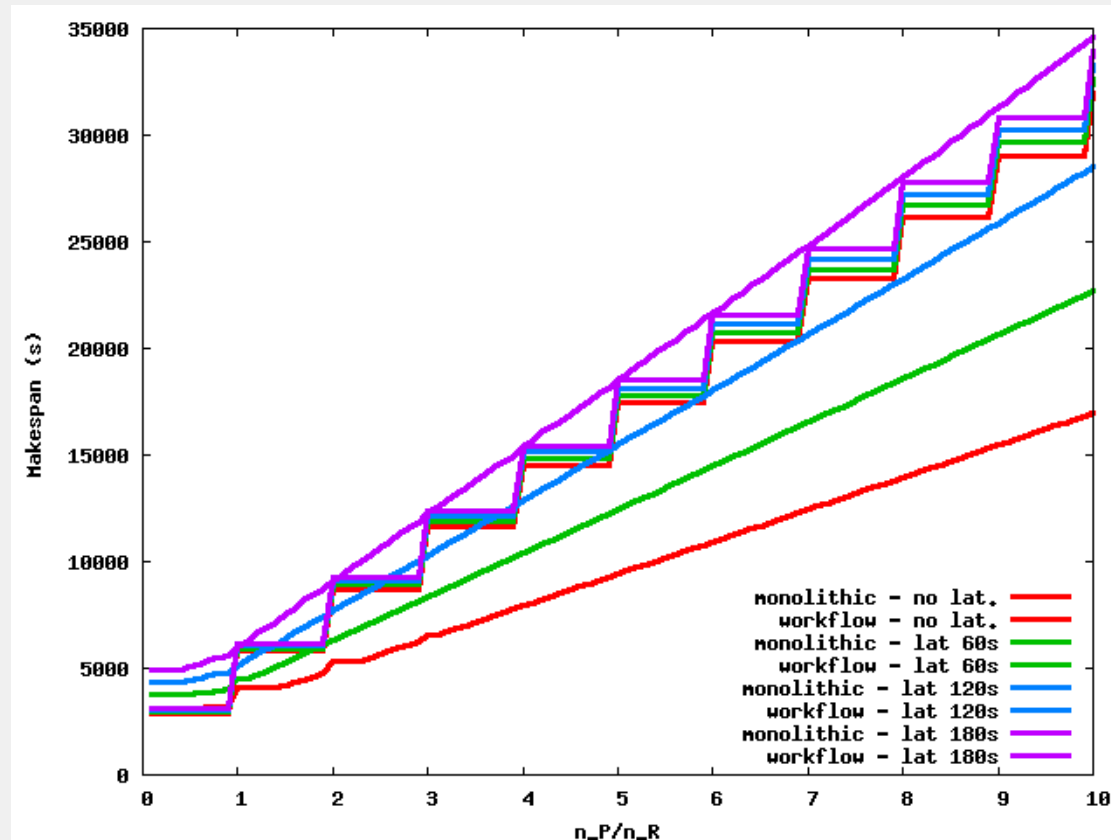


- Workflow more sensitive to latency



# Results: parameter sweep

- With data transfers and latency



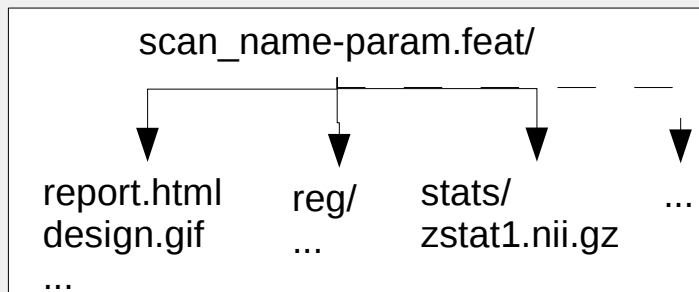
Latency increases

- Workflow outperforms monolithic for realistic latency values



# Output organization: *problem*

- Regular *feat* output



- Directory structure matches experiment logic
- Easy file retrieval

- Workflow output (as in MOTEUR)

## thresh\_zstat1\_nii\_gz

- [/grid/vlmed/tristan/feat\\_workflow\\_results/thresh\\_zstat-1218114106910154](#)
- [/grid/vlmed/tristan/feat\\_workflow\\_results/thresh\\_zstat-1218114118923193](#)

## design\_mat

- [/grid/vlmed/tristan/feat\\_workflow\\_results/design-121810923151167915377](#)
- [/grid/vlmed/tristan/feat\\_workflow\\_results/design-121810924765259415377](#)

## stats\_smoothness

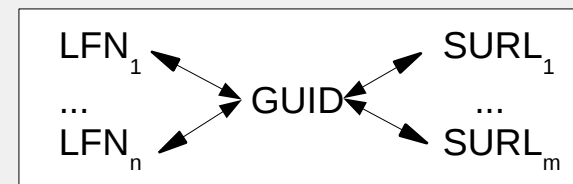
- [/grid/vlmed/tristan/feat\\_workflow\\_results/smoothness-1218113661354491](#)
- [/grid/vlmed/tristan/feat\\_workflow\\_results/smoothness-1218113643548207](#)

- Automatically generated file names
- Provenance info available



# Output organization: *constraints*

- **Meaningfulness**
  - Easily retrieve a particular file
  - Associate it to the input parameters
- **Reusability**
  - Components among workflows
  - Workflows among users
- **Grid-awareness**
  - Distributed storage
  - File replication, move
  - LFN change



# Output organization: *existing approaches*

	Meaningful	Reusable	Grid-aware
Components produce GUIDs	✗	✓	✓
Provenance GUID browsing	✗	✓	✓
Result LFNs function of inputs	✓	✗	✓
LFN annotation with metadata	✓	✗	✗



# Conclusions

- Description of *feat* workflow feasible
  - For a specific use-case (e.g. fixed number of EVs)
  - Requires a tiresome analysis
- Workflow performance evaluation
  - Execution time reduction for parameter sweep
  - Data transfers and latency prevail for job farming
- Output organization
  - Should be grid-aware, reusable and meaningful
  - Components-, workflow- and execution-independent
- Sharing complex workflows is still difficult
  - Use-case specific implementation

# Thanks for your attention!

*Downloads, demos and videos available from  
<https://pc-vlab18.science.uva.nl:8080/vbrowser/>  
(and on my laptop...)*

## *Acknowledgement:*

- **AMC, University of Amsterdam**
  - S.Olabarriaga, K. Boulebiar, A. van Kampen
  - A. Nederveen, M. Caan, S. Gevers, R. Soleman, D. Veltman
- **Informatics, University of Amsterdam**
  - P. de Boer, A. Belloum
  - R. Belleman, R. Bakker
  - S. Marshall, M. Roos
  - Prof. Dr. L.O.Hertzberger
- **SARA Supercomputing Services**
  - M. Bouwhuis, J. Engelberts, Ron Trompert, [grid-support@sara.nl](mailto:grid-support@sara.nl)
- **National Institute for Nuclear Physics and High Energy Physics (NIKHEF)**
  - J.J. Keijser, D. van Dok, J. Templon, [grid-support@nikhef.nl](mailto:grid-support@nikhef.nl)



**<http://www.vl-e.nl/>**