Using Relations Between Fluid Pressure and Mean Effective Stress to Accelerate Iterative Coupled Simulation

by

Øystein Pettersen, CIPR

FORCE / JCR Workshop on Coupled Modelling,
NPD, 21.-22. Nov. 2006

Acknowledgements:
Support from BP / Valhall and the Visage team
(V.I.P.S. Ltd.) is greatly appreciated

Outline

- ***Introduction***
- ***Some theoretical considerations***
- ***Procedure description***
  - ***Handling of simple and complex cases***
- ***Examples & results***

Notation

Reservoir state: $\Sigma = \Sigma_F + \Sigma_R$,

$\Sigma_F = \boldsymbol{u}, p_f, S_l, \ldots$: Flow state

$\Sigma_R = \boldsymbol{\sigma}, \boldsymbol{\varepsilon}, \boldsymbol{\zeta}, \ldots$: Rock state

PV: (Cell) Pore volume

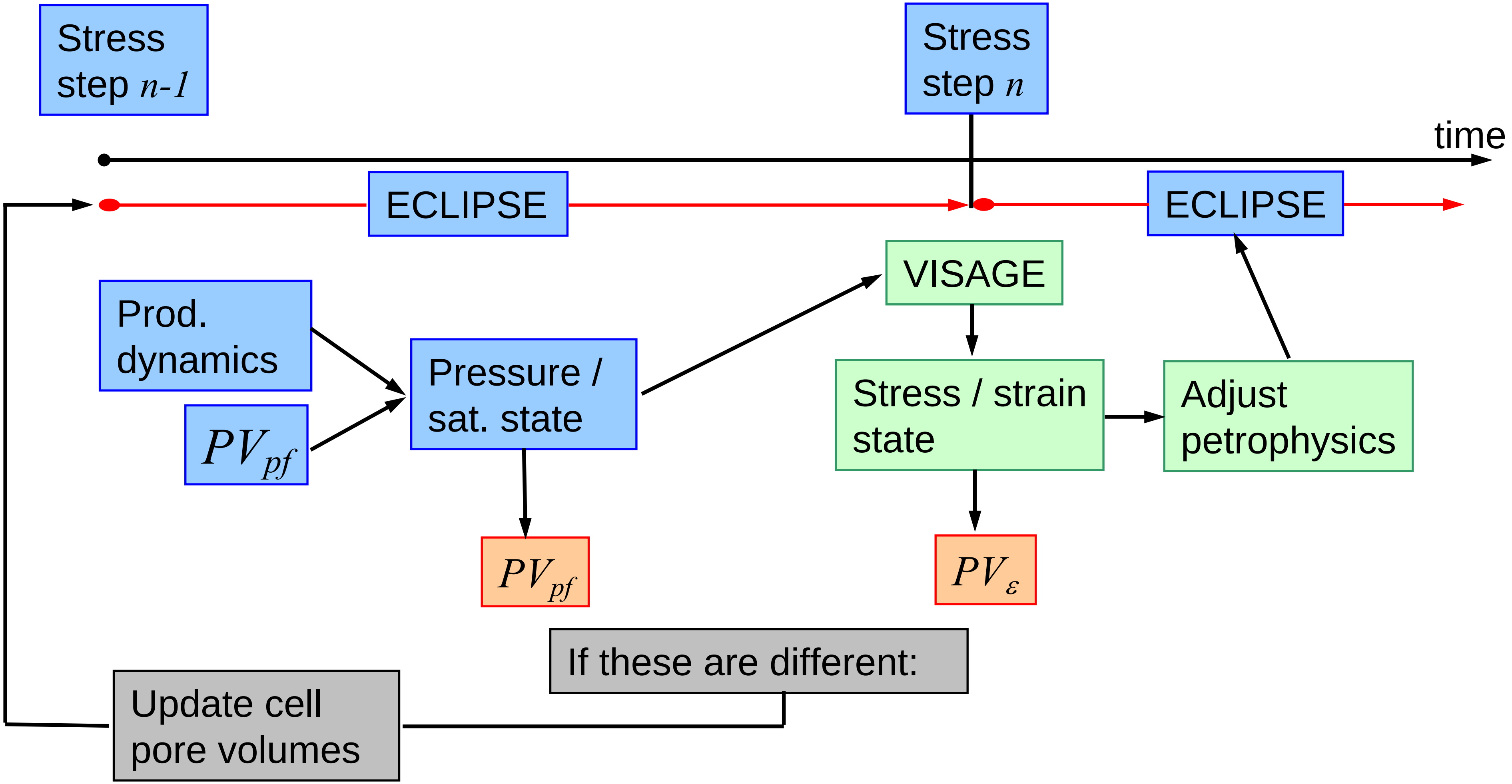
$m(t) = \dfrac{PV(t)}{PV_{init}}$ Pore Volume multiplier

$m_{pf}(t)$: Computed from fluid pressure (table look-up)

$m_{\varepsilon}(t)$: Computed from volumetric strain

$$m_{\varepsilon}(t) = e^{-\Delta\varepsilon_{\mathrm{vol}}}$$

Coupling Scheme – Iterative



Classic iteration

To solve the problem

$f(x) = x$

by iteration,

set $x^0 = c_0$ (iteration initialiser)

$x^n = f(x^{n-1})$ until $|x^n - x^{n-1}| <$ tol.

Intuitively, when x^0 is closer to x,
fewer iterations will be needed.

Technical description of solution procedure

To solve for reservoir state at stress step n:

Solve rock mech. system $\boldsymbol{f}(\Sigma_R) = \boldsymbol{0}$ (*), subject to BC and iteration initialiser,

$\Sigma_{init}^{n} = (\Sigma_R^{n-1}, \Sigma_F^{n})$ where

Σ_F^{n} was delivered by the flow simulator

In an explicit stress step, the solution to (*) is found by *solver iterations.*

To ensure accurate reservoir state, cell pore volumes as computed by flow sim are compared to those computed by rock mech sim:

$$\text{If } \left\| PV(\Sigma) - PV(\Sigma_F) \right\| > \text{tol},$$

$$\text{set } PV(c_i) = PV_\varepsilon(c_i) \text{ in all cells } c_i$$

Repeat stress step until convergence *(pore volume iterations)*

The iteration initialiser

The stress step initialiser is dependent on flow sim computed compaction through

$$\Sigma_F^n = (p_f, m_{pf}, S_l, \ldots)^n$$

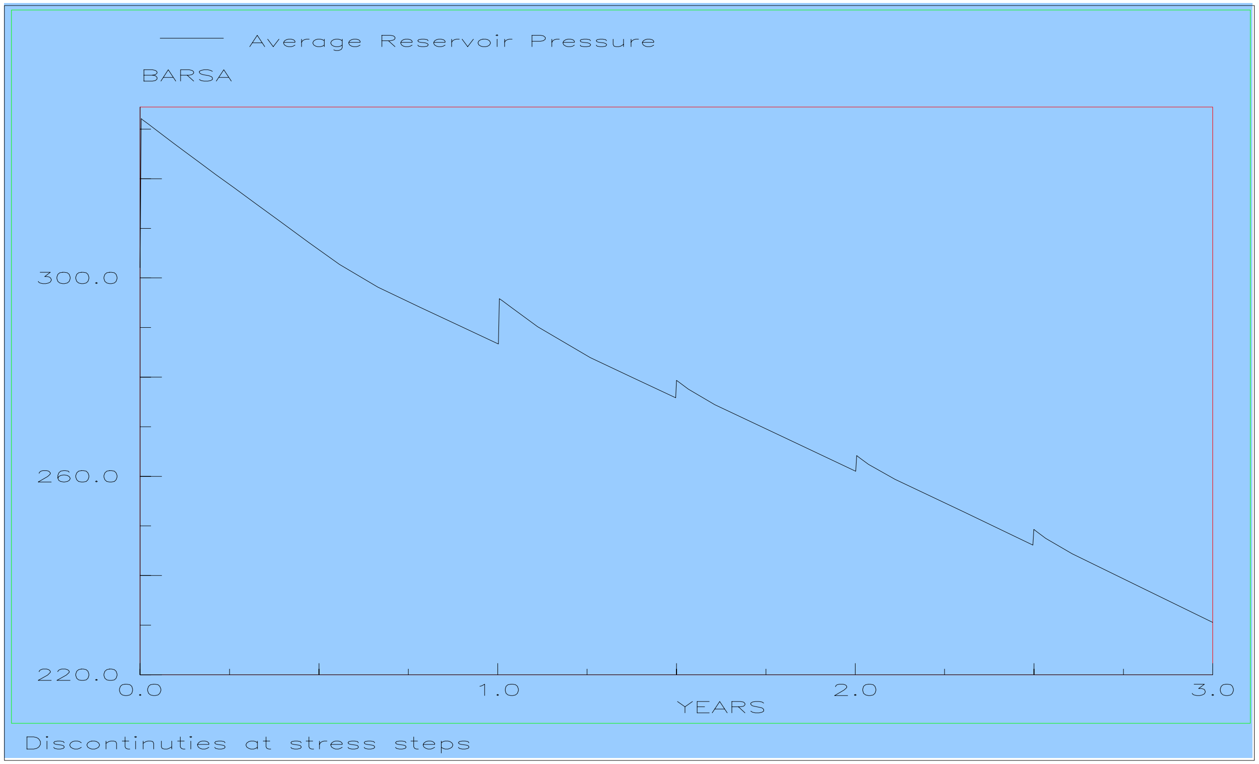
Hence the cell pore volumes computed by the flow simulator are predetermined by the input PV-mult table (”Rock Table”)

By pore volume iterations, the correct reservoir state will eventually be found, but intuitively, the number of iterations needed will depend on how good the starting point is.

Flow sim PV-update ⇒ altered p_f-field ⇒ altered PVs
⇒ often low convergence rate

Fluid pressure and pore volume updates

p_f between stress steps is a function of input Rock Tables.
If these cannot adequately describe the reservoir state,
p_f will only be correct at stress steps,
and discontinuous due to PV-correction.



Discontinuties at stress steps

Quest for the Golden Chalice

Construct a modified material description
(PVM-tables)
such that cell pore volumes and fluid pressure
are accurately computed already by the flow simulator

Static and dynamic variables

Reality: $m = m(p')$ $(p'$: mean effective stress)

Flow simulator: $m = m(p_f)$

$p' = p'(BC, p_f, \sigma, \ldots)$

$p_f = p_f(m, \text{Process}, \ldots)$ (Process: Well positions & rates)

$\sigma = \sigma(\text{Matr. def.})$ (Properties, distribution)

Hence

$p' = p'(BC, p_f, \text{Matr. def.}, \text{Well pos. \& rates})$

$\Rightarrow$

$m = m(p')$

$\quad = m(BC, p_f, \text{Matr, def., Well pos \& rates})$

Static data: BC, init. matr. props, matr. distribution, well pos

Dynamic data: p_f, well rates, matr. props

Splitting

m-function is split into two parts (static and dynamic):

$$\begin{aligned} m &= m(BC,\, p_f,\, \text{Matr, def., Well pos \& rates}) \\ &= f_S\,(\mathbf{x},\, p_f;\ \text{BC, Init. matr. def., Well pos.}) \\ &\quad + f_D\,(\mathbf{x},\, p_f,\, t\ ;\ \text{Dyn. matr. props, Well rates}) \end{aligned}$$

By its nature f_S is a function of the reservoir parameters.
At each stress step, changes in m are taken care of by f_D.

f_D changes much more slowly than Σ.

With $\Delta f_D^n = |\, f_D^n - f_D^{n\text{-}1}\, |$,
if Δf_D^n is small, only a few PV-iterations if any will be needed.

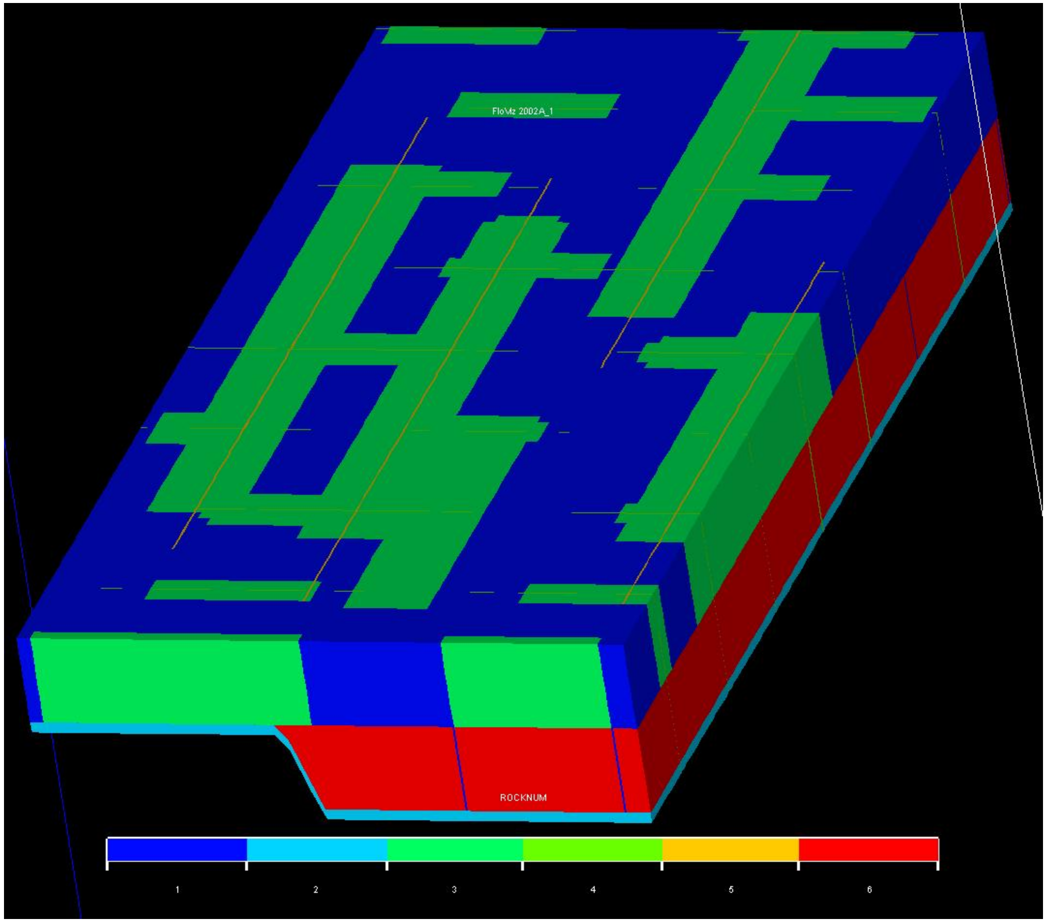
Test case 1 – Single Material (SM)

- Box-shaped reservoir comprised of a single material
  - ➢ Moderately weak high perm. sandstone (Brent)
  - ➢ Critical State with initially vanishing ellipse axes
    - ✓ load enters plastic region immediately
    - ✓ horizontal unloading lines (permanent deformation)
  - ➢ Depletion or voidage replacement (no global unload)
- Base case:
  - – Row of injectors along western edge
  - – Row of producers along eastern edge
- ”Standard” modelling of over / under / side-burdens (MC)
- Simple, but essential to understand and classify relationships

Test case 2 – Multiple Material Chalk (MMC)

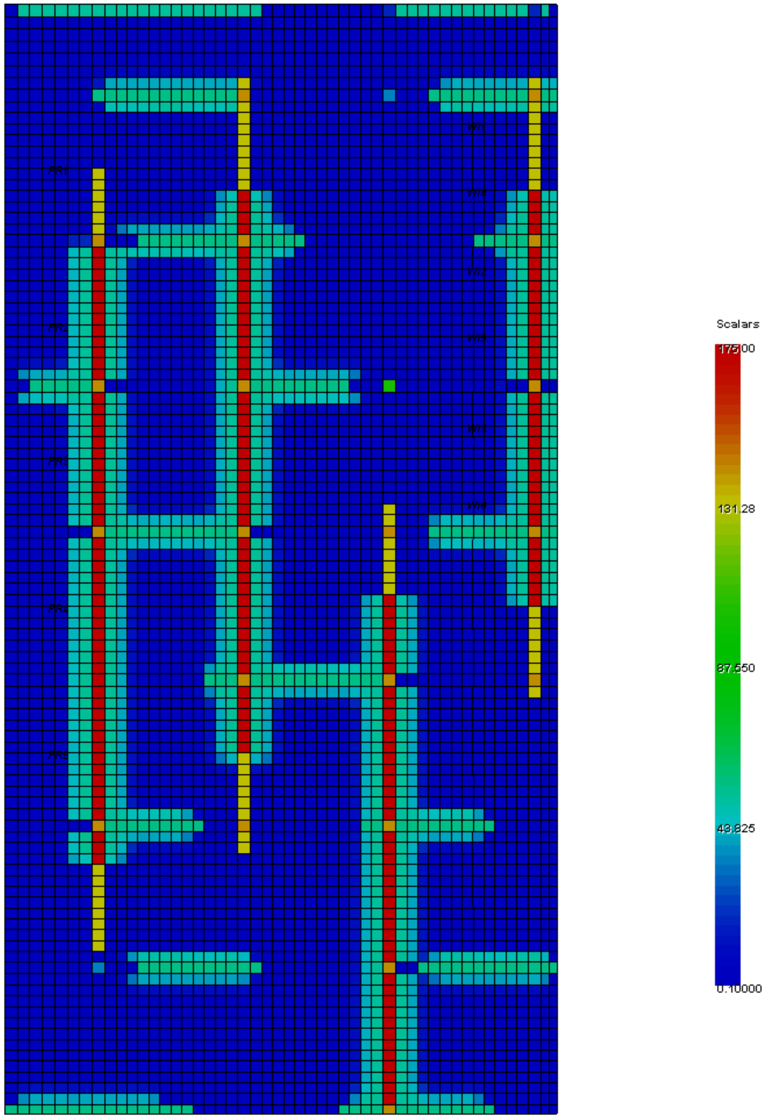
- Reservoir comprised of heterogeneous and anisotropic soil, essentially grouped as six different materials
  - Matrix (Chalk)
    - NGI chalk model with Valhall parameters
    - Swelling
  - Fractures (MC)
    - Different description for EW fracs and NS fracs
    - Fracture closure (perm. reduction w. load)
  - Transition zones (Chalk)
    - Perm. reduction
  - Pinchout zone (Chalk)
  - Hardground (MC)
- ”Standard” modelling of over / under / side-burdens (MC)

Case MMC



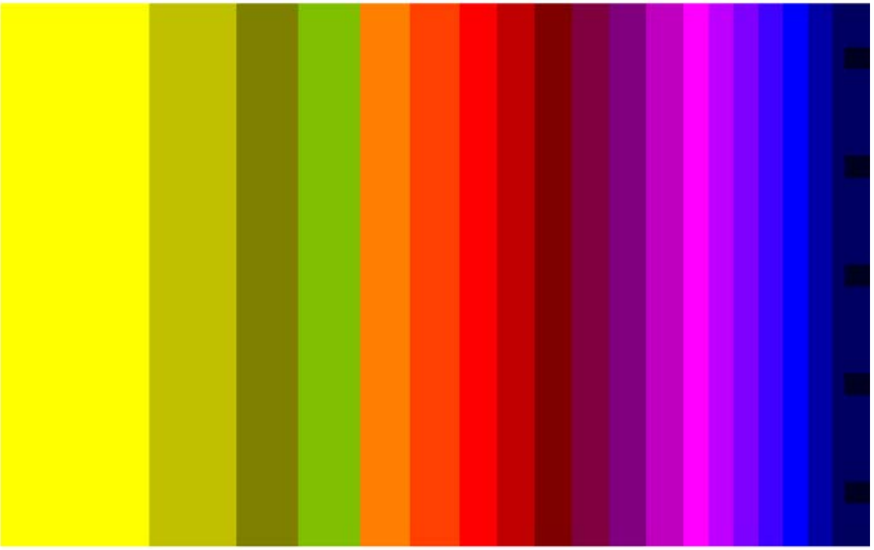
Materials distribution

Model: RMAB : 30/9/1982
Timestep: INPUT : 0 days
CellData Scalar: y-permeability
Min: 0.1 Max: 175 dimensionless

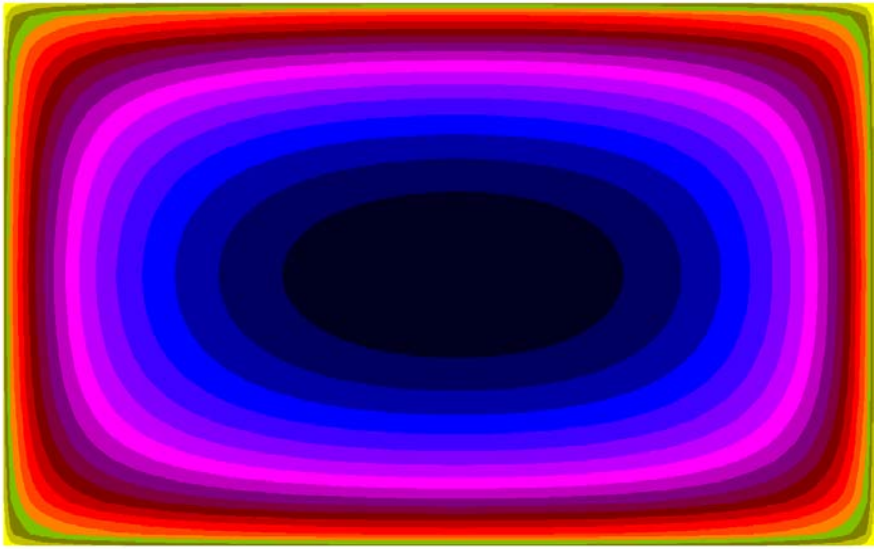


K_y, top layer

Base SM run – BC influence



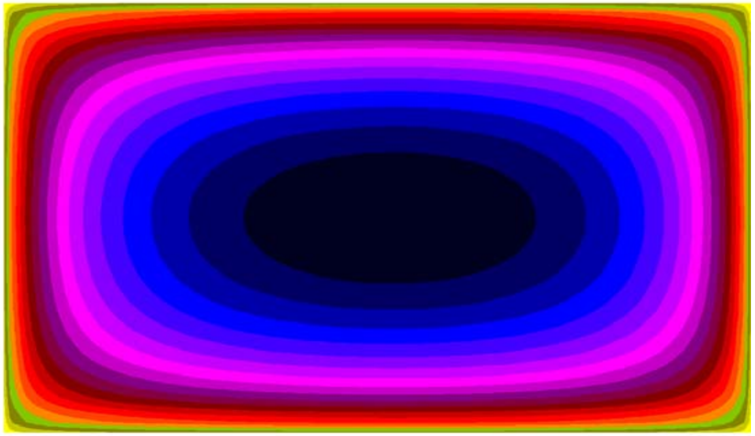
p_f (from Eclipse)



m_ε (from Visage)

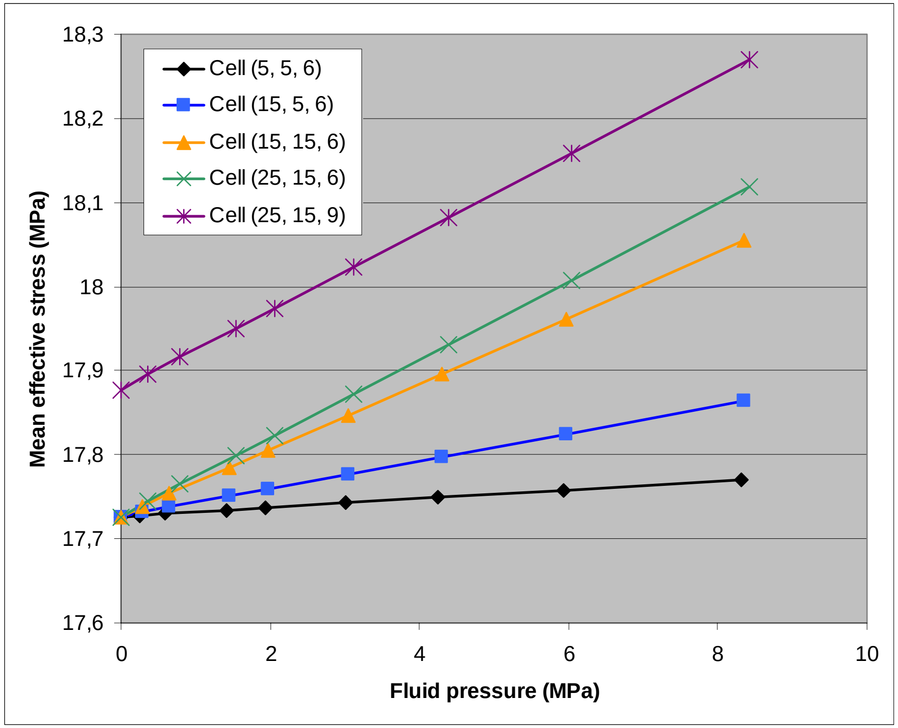
Although $p_f(\boldsymbol{x})$ is very unsymmetric, m_ε is almost symmetric. This is an indication that BCs have a stronger influence than the process, and that Δf_D is indeed small.

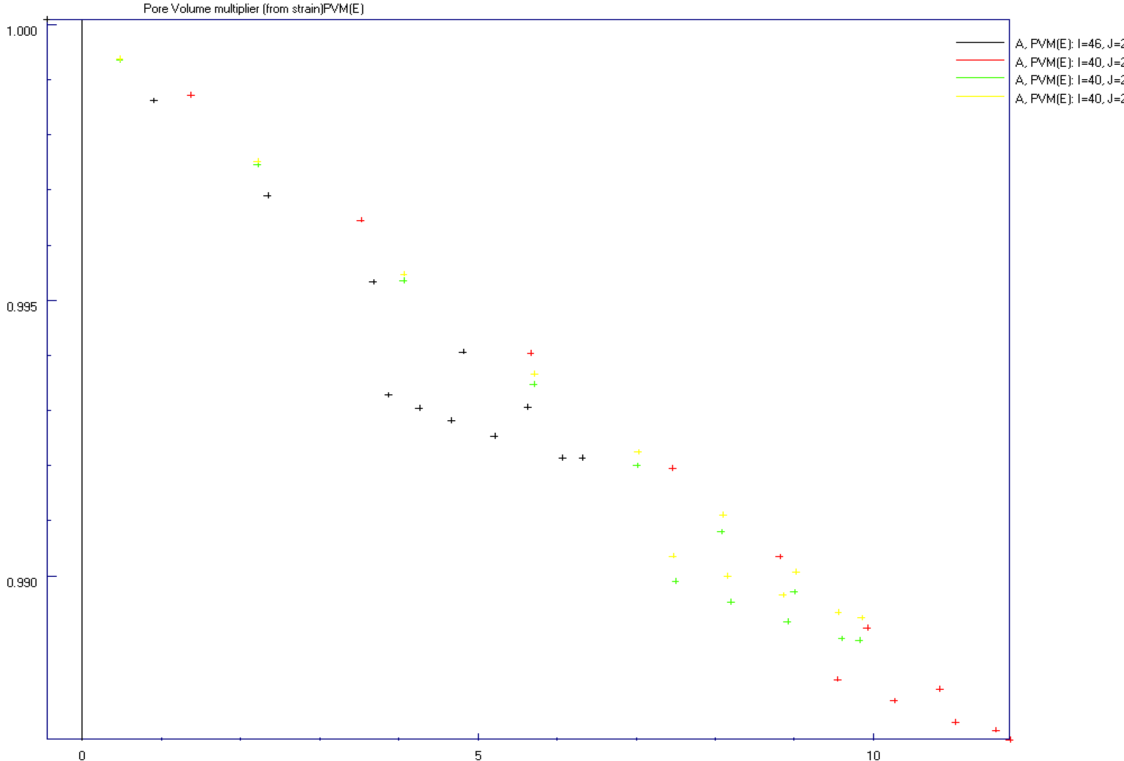
Localized behaviour



m_{ε} – contours case base SM.
Systematic – clearly BC governed

$p'(p_f)$ in some single cells
base SM (down left); MMC (down right)

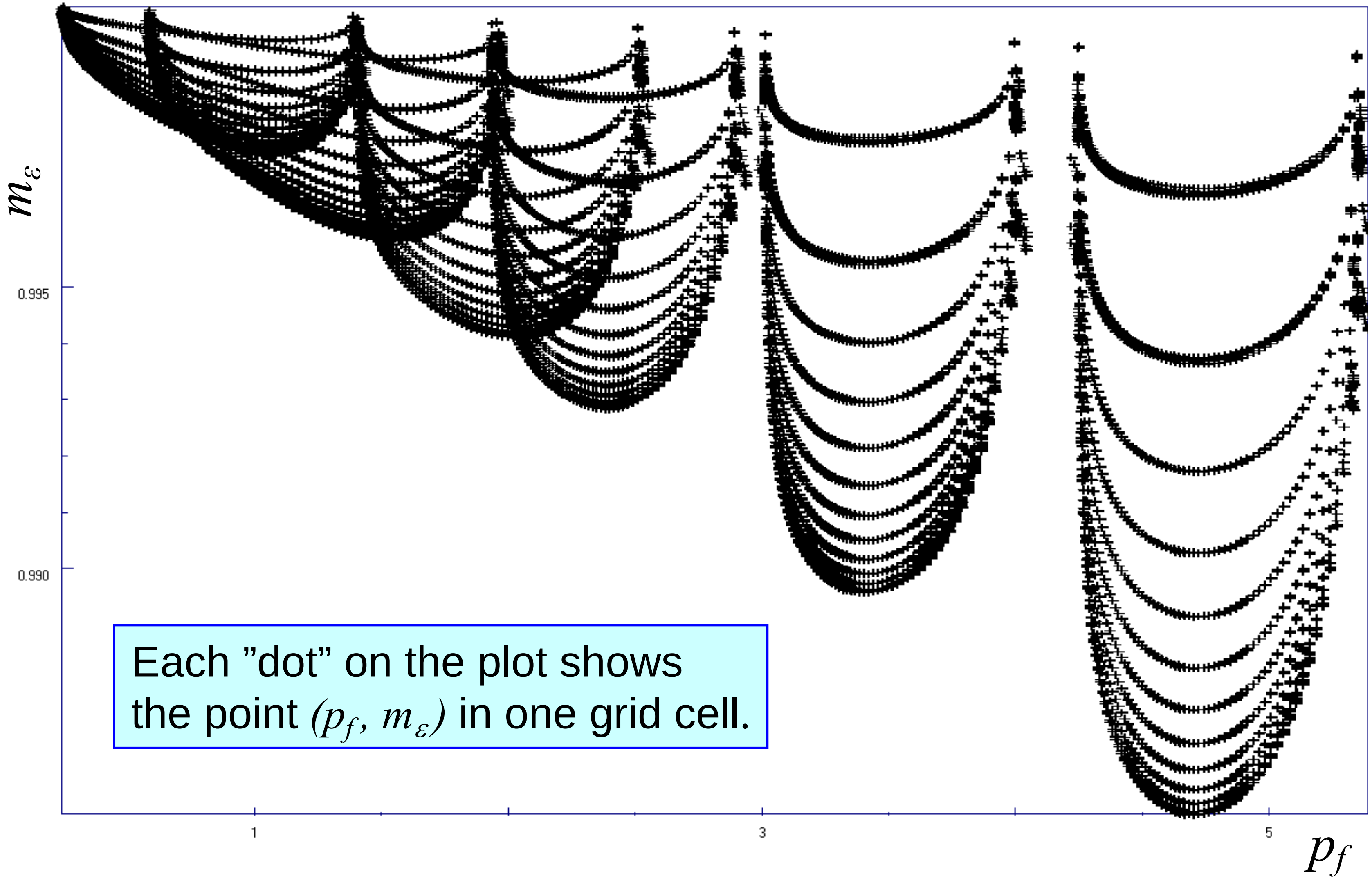




Construction process

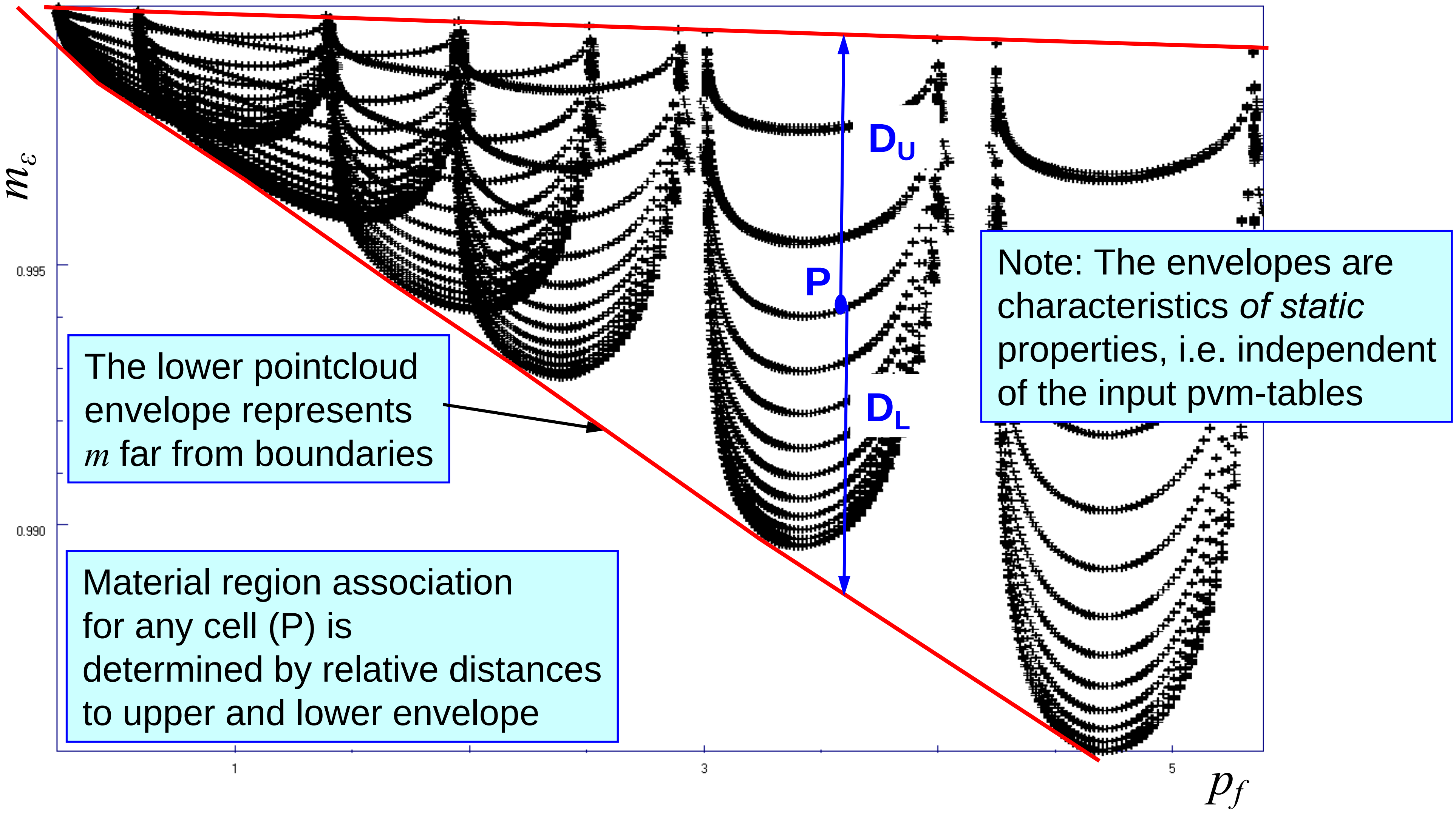
- Assume it is possible to construct a set of *local* compressibility functions $m(p_f; \boldsymbol{x})$, where $\boldsymbol{x}$ takes the role of a parameter specifying the validity range for the function in question.
- How to determine $\boldsymbol{x}$ (in practice a set of grid cells), and the associated function (pvm-table)?
- Construction process is based on analysis of a pointcloud of (p_f , m_ε)-pairs, obtained from a Tuning run.
- The Tuning run should contain at least three stress steps, and cover the entire load range of interest
- The Tuning run can be run in explicit coupled mode

Pointcloud with 7 stress steps (SM)



Each ”dot” on the plot shows the point (p_f, m_ε) in one grid cell.

Pointcloud analysis

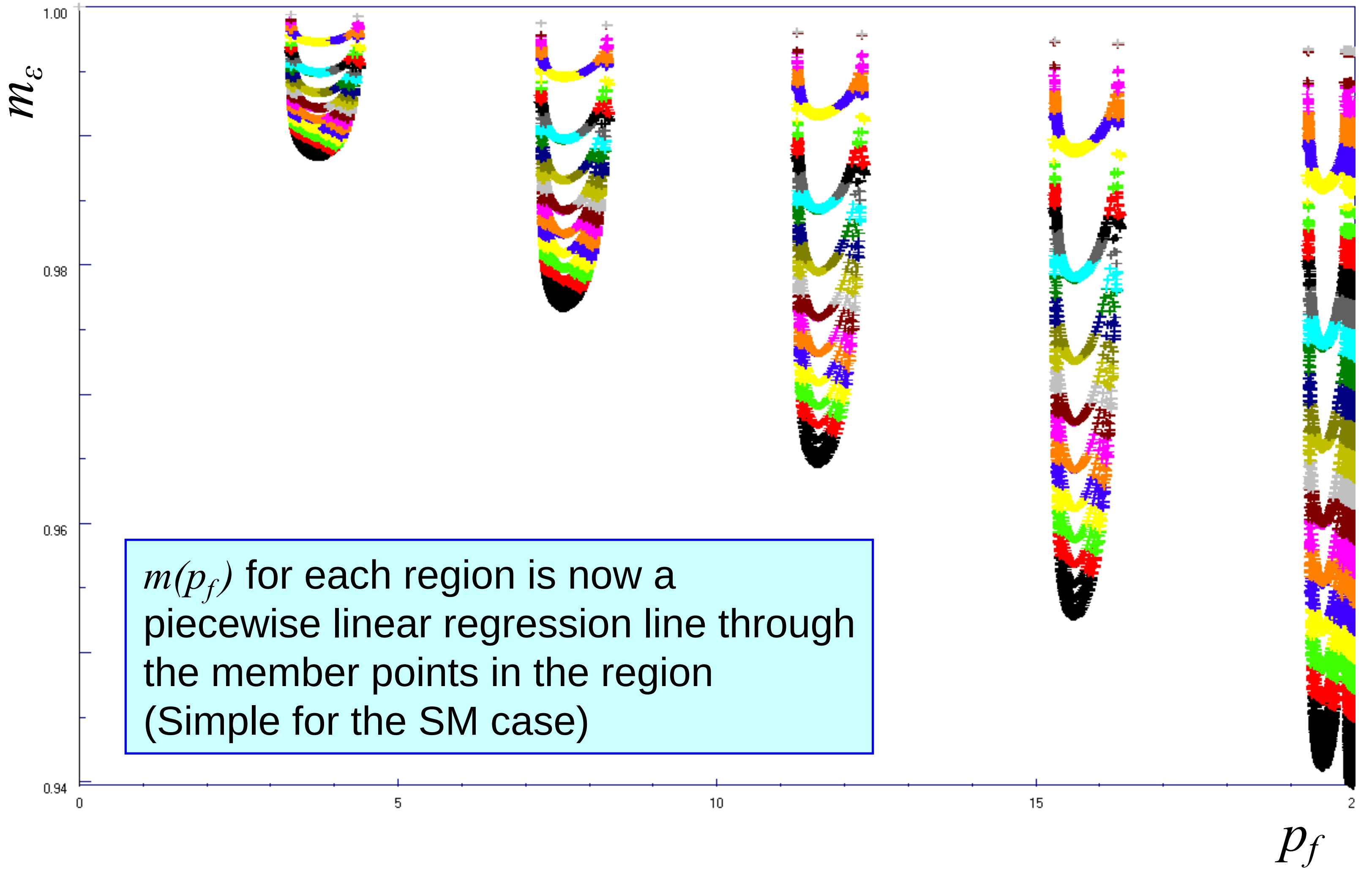


The lower pointcloud envelope represents *m* far from boundaries

Material region association for any cell (P) is determined by relative distances to upper and lower envelope

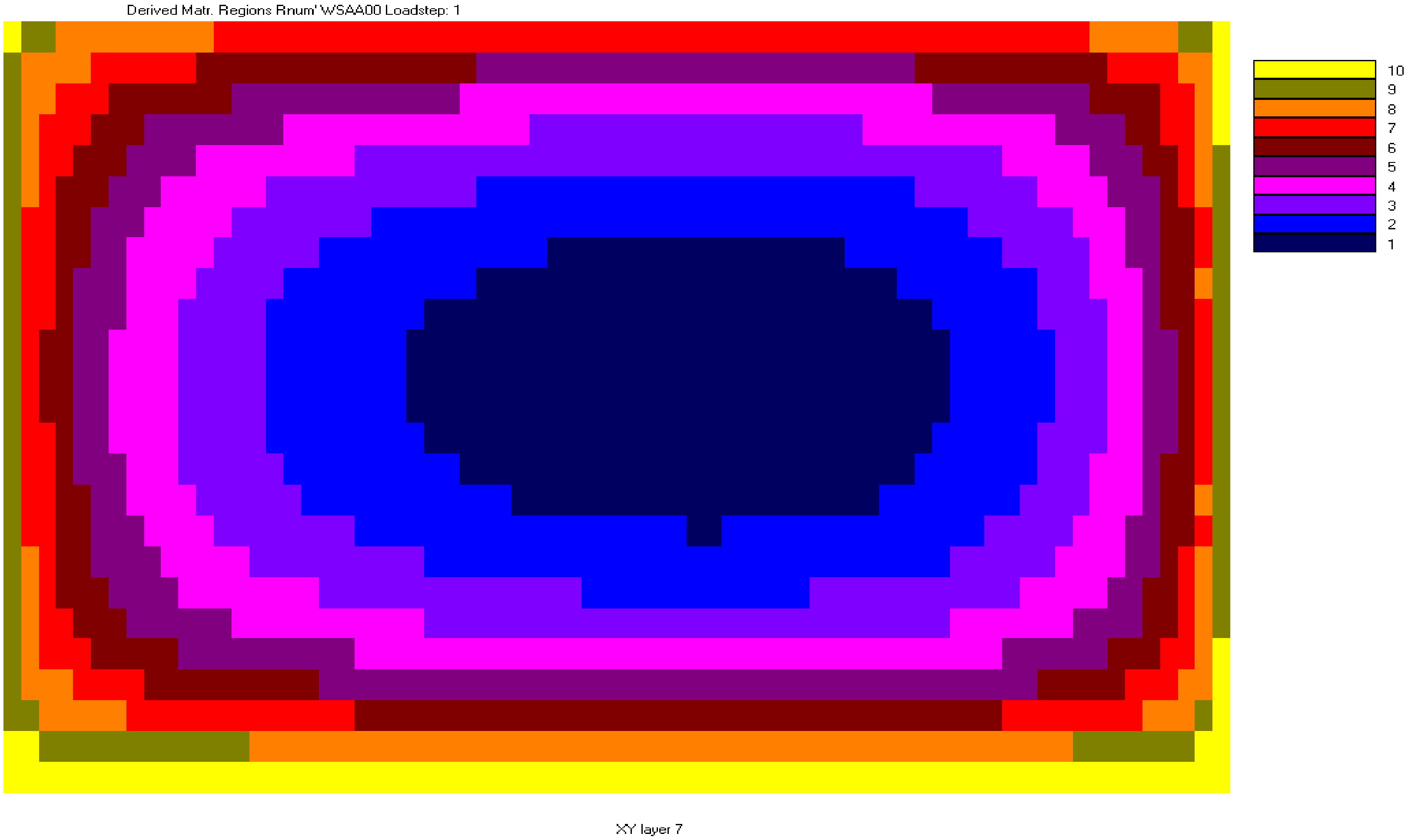
Note: The envelopes are characteristics *of static* properties, i.e. independent of the input pvm-tables

Assigning material regions to grid cells (SM)



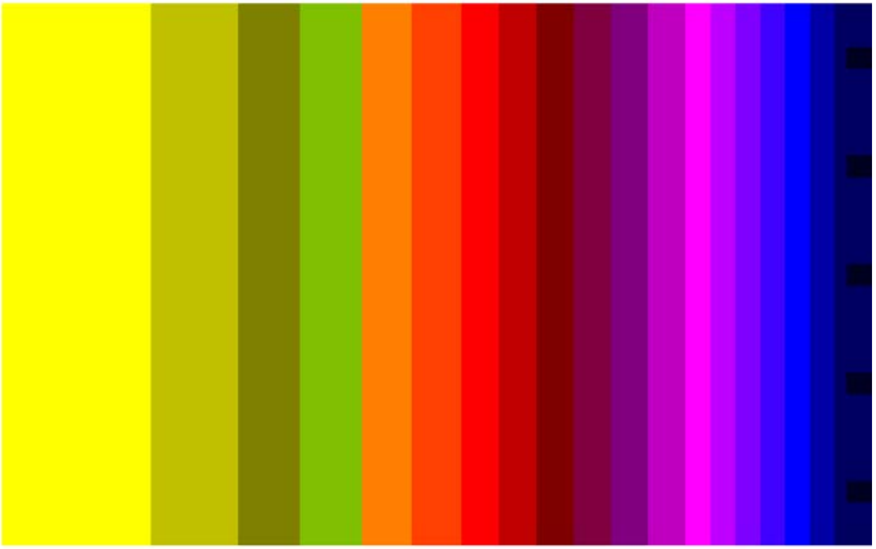
$m(p_f)$ for each region is now a piecewise linear regression line through the member points in the region (Simple for the SM case)

Ex. Grouping in Material Regions, XY View

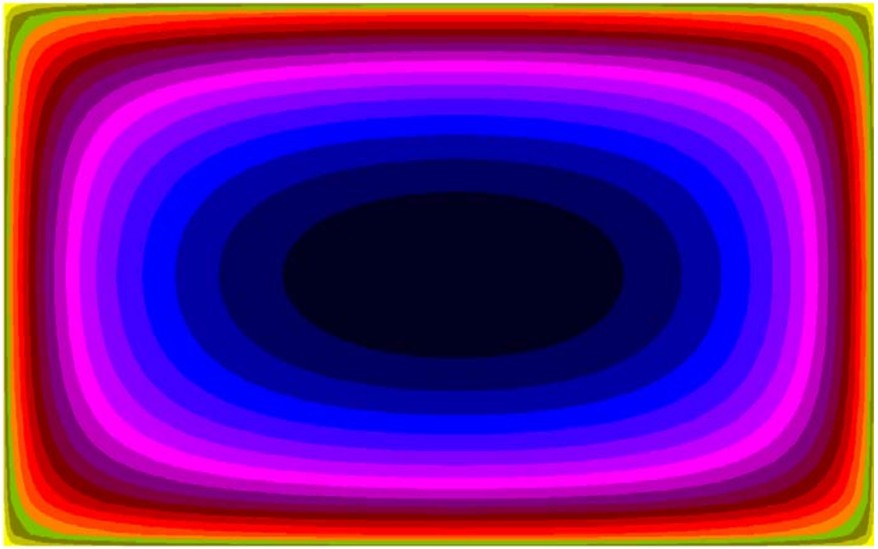


Results from base SM run

Using “standard” pvm-tables



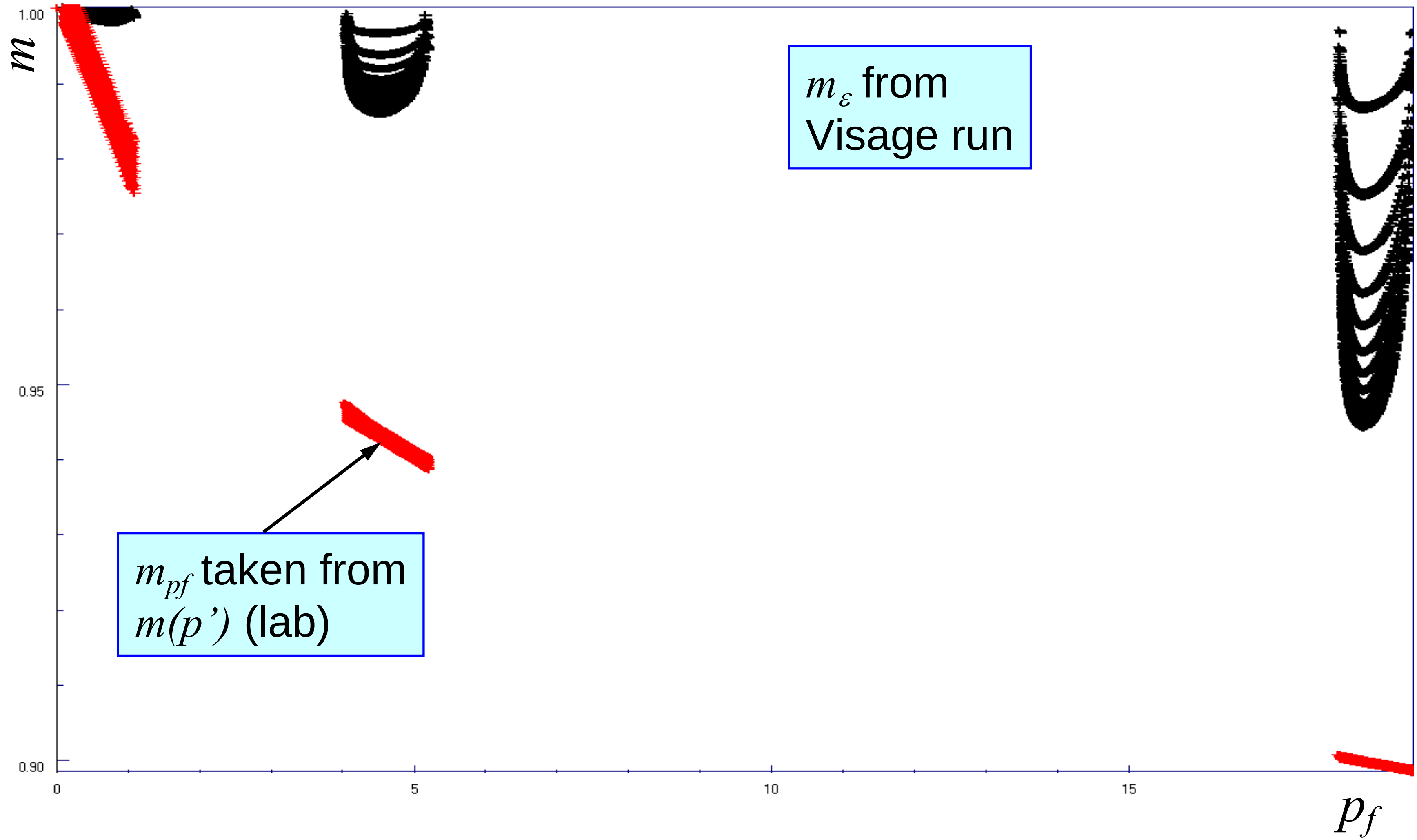
m_{pf} (from Eclipse)



m_{ε} (from Visage)

Note: Visage run explicit coupling
Although distribution is qualitatively correct, level may be wrong

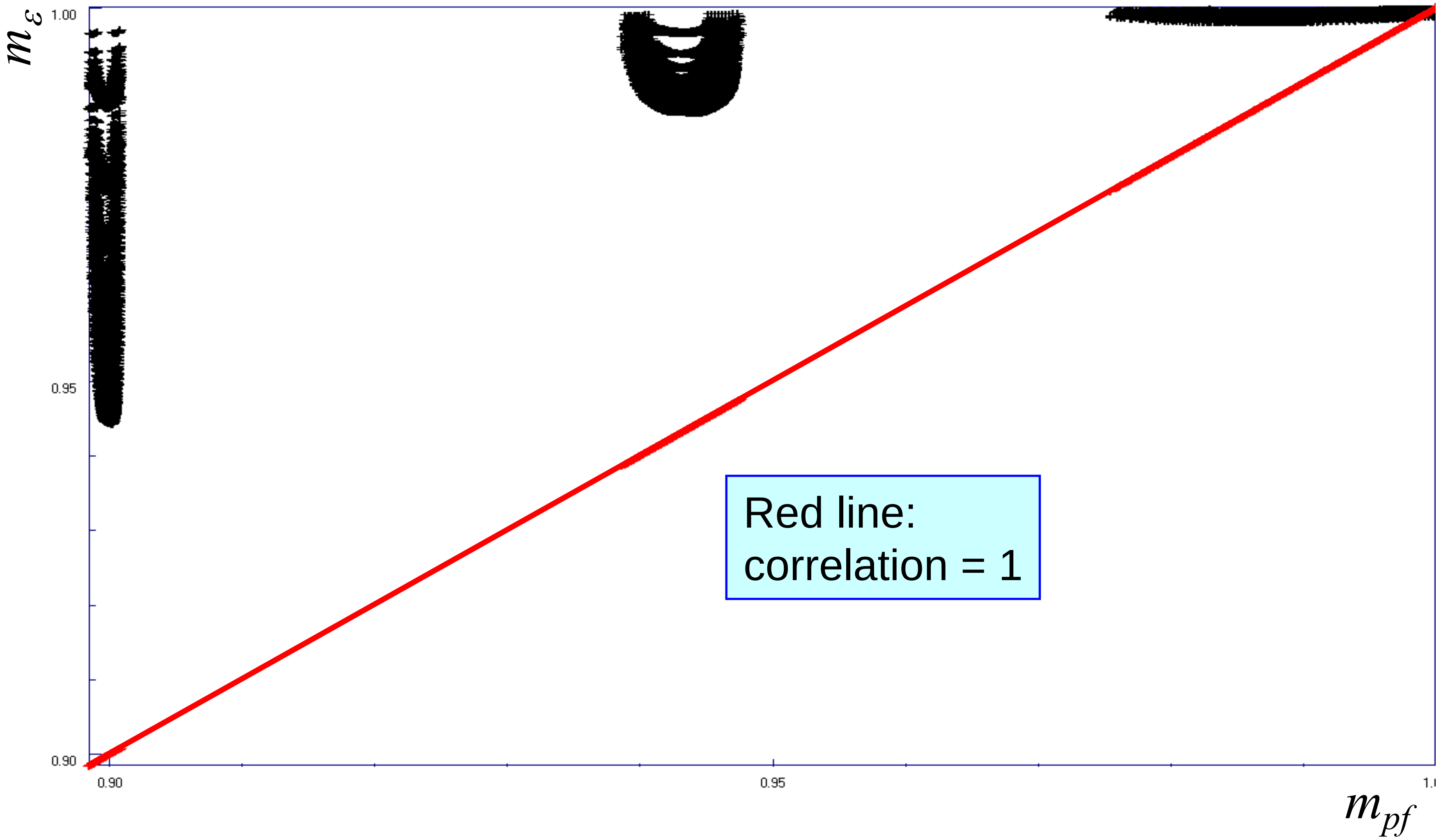
Tuning run SM, 3 stress steps, m vs. p_f



m_ε from Visage run

m_{pf} taken from $m(p')$ (lab)

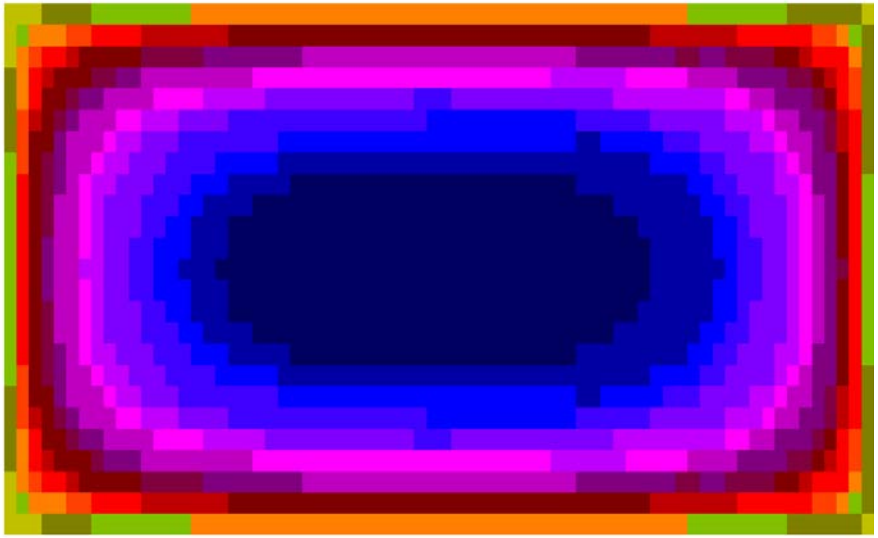
Tuning run SM, correlation m_ε vs. m_{pf}



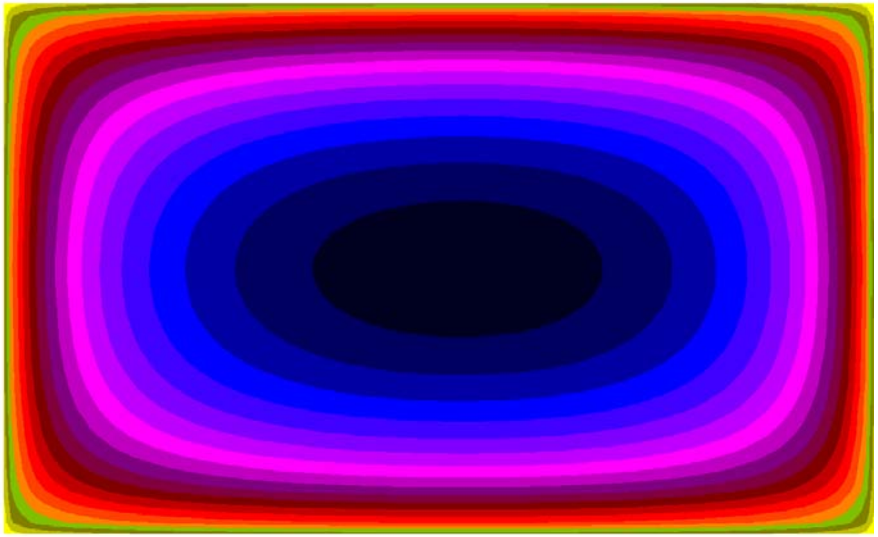
Red line:
correlation = 1

Results from base SM run

Using new “pseudo” material regions and pvm-tables
Stress step 4, reservoir layer 2



m_{pf} (from Eclipse)



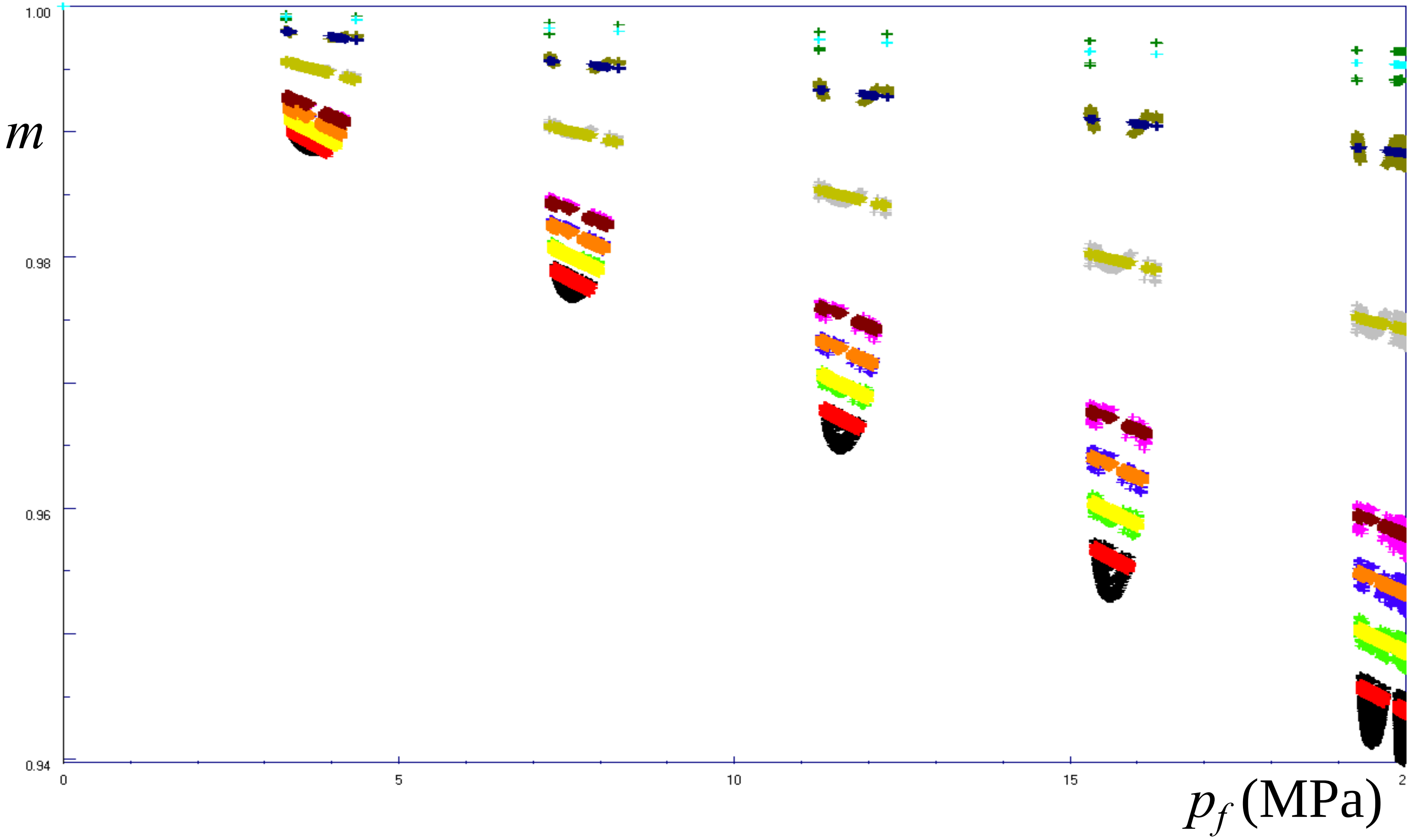
m_{ε} (from Visage)

Note: Visage run explicit coupling
Both distribution and level is now OK

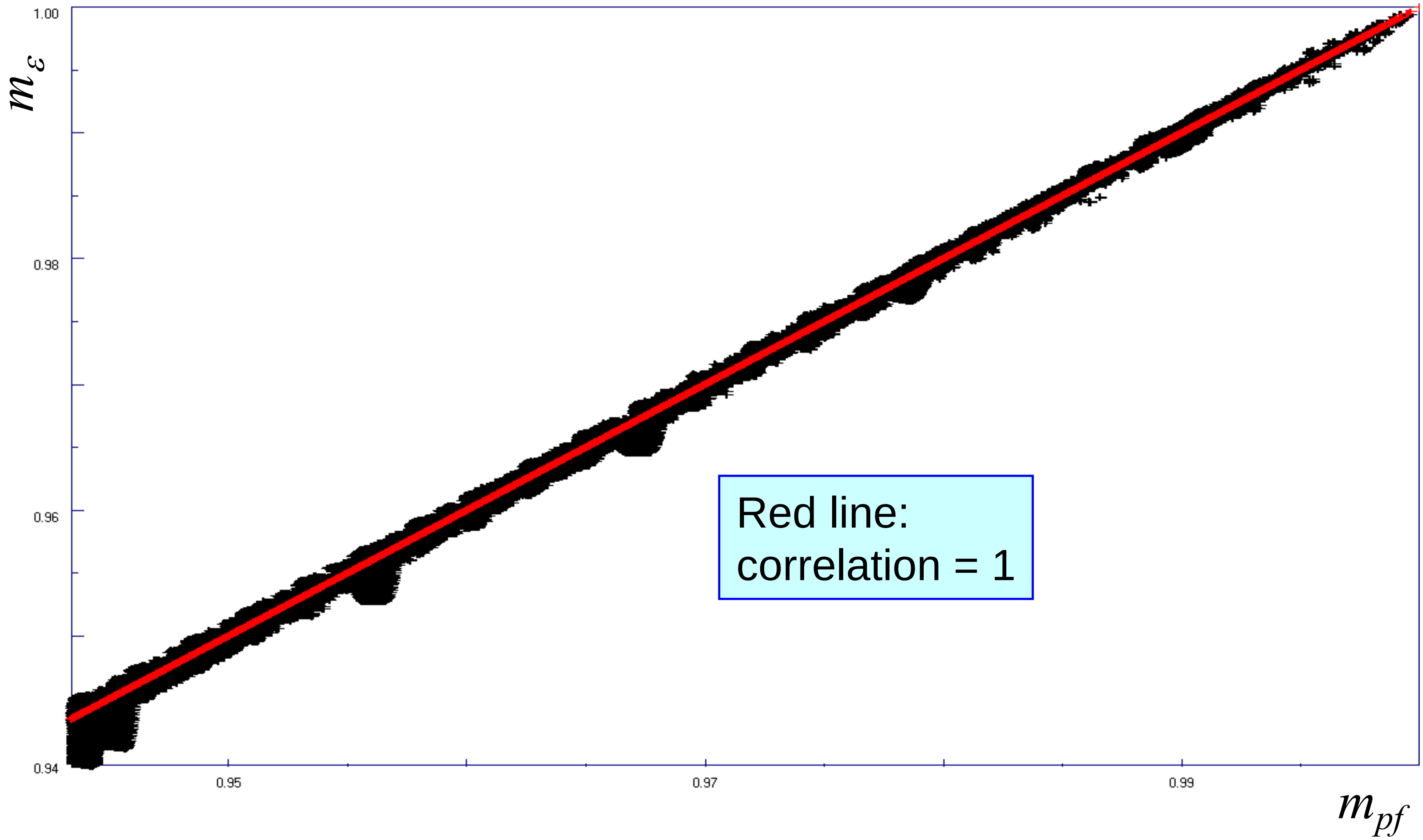
Results from base SM run

Using new “pseudo” material regions and pvm-tables

m_ε and m_{pf} vs. p_f (some matr. regions shown)

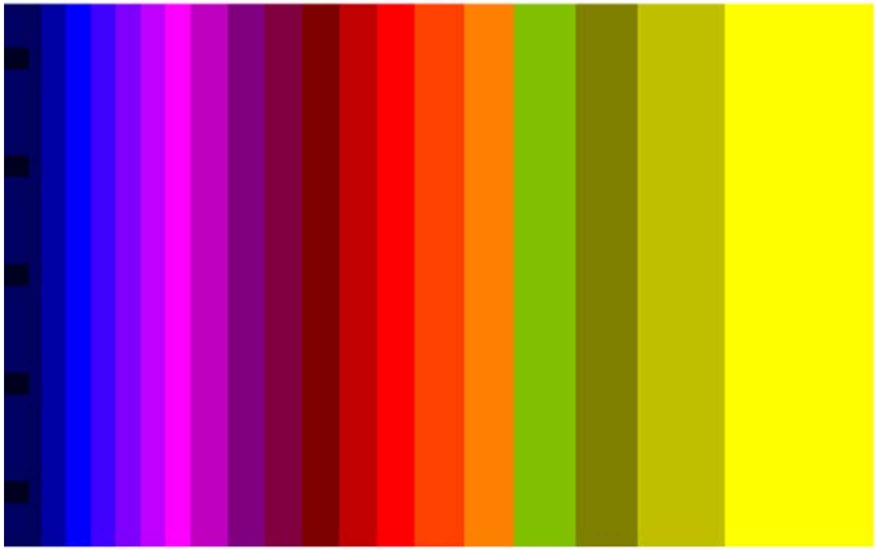


Base SM, correlation m_ε vs. m_{pf}

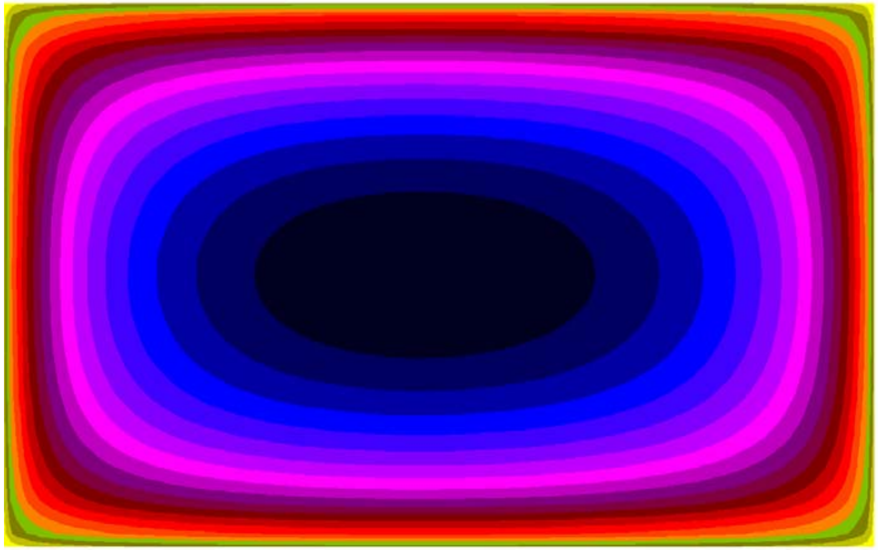


Red line:
correlation = 1

Sensitivity test 1: Swap injectors and producers



p_f (from Eclipse)

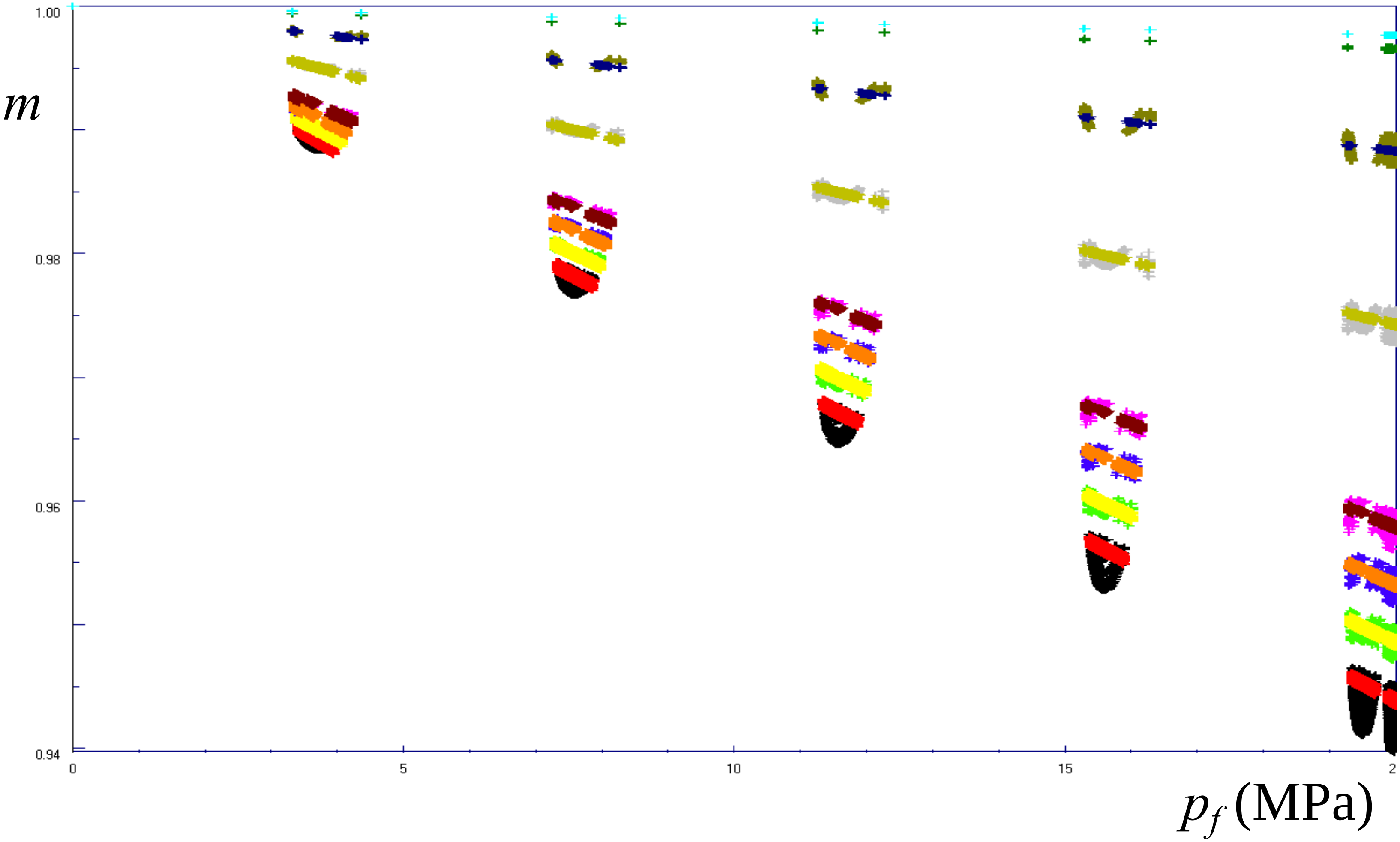


m_ε (from Visage)

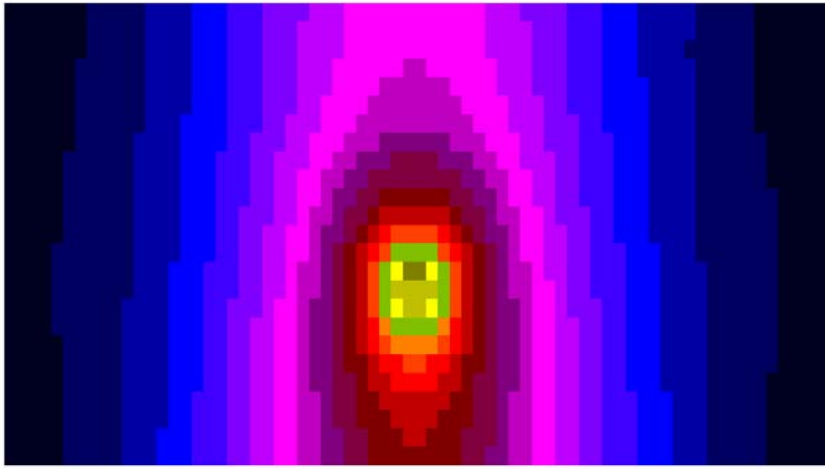
Note: All the sensitivity tests were run with the material regions and pvm-tables determined in the base SM. (No new tuning run)

Sensitivity test 1: Swap injectors and producers

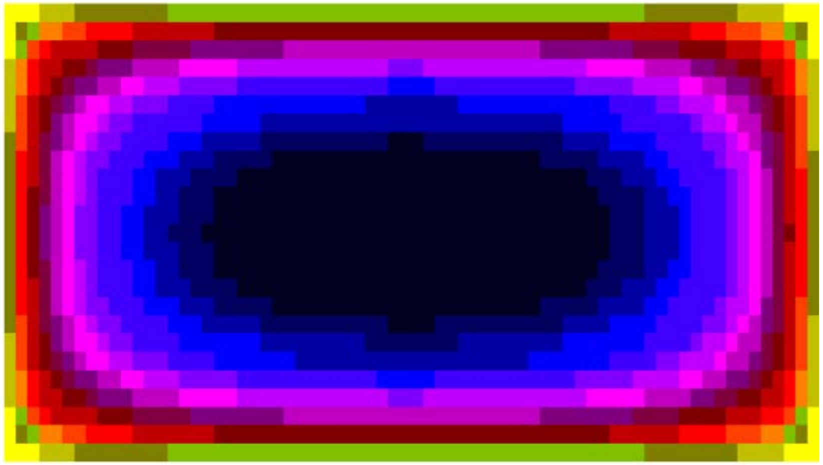
m_ε and m_{pf} vs. p_f (some matr. regions shown)



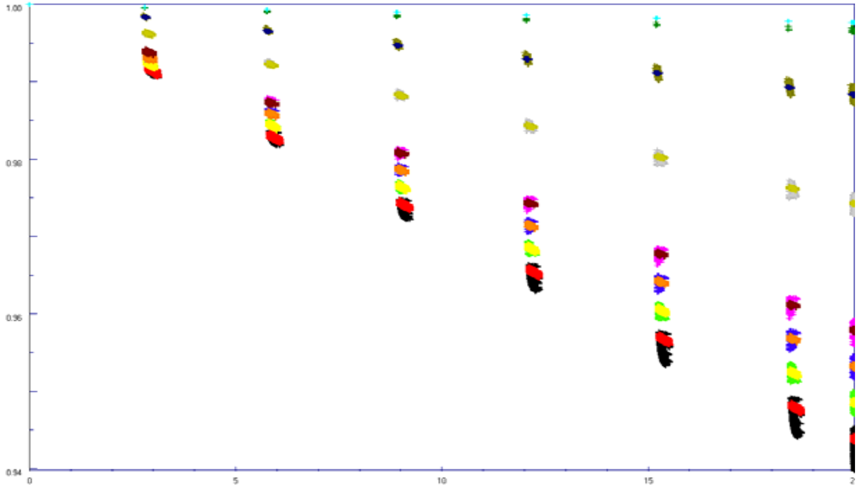
Sensitivity test 2: Unsymmetric 5-spot



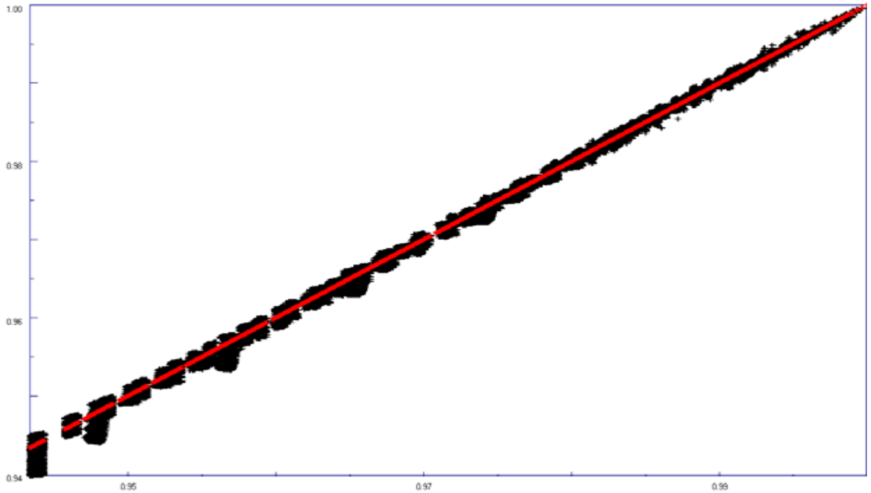
p_f (from Eclipse)



m_{pf} (from Eclipse)



m_ε and m_{pf} vs. p_f



Correlation m_ε vs. m_{pf}

Conclusions Simple Single Material Case

- The pseudo material regions (Eclipse ROCKNUM) are clearly defined and easy to construct
- Associated pvm-functions (Rock Tables) are also well defined
- Constructed ROCKNUM / ROCKTABs are independent of the initial pvm-function used in the tuning run
  - May require “non-unrealistic” data
- The pseudo ROCKNUM / ROCKTAB are **robust**, once constructed they behave (surprisingly) well also for alternative well configurations
- Computed compaction by Eclipse matches (almost) perfectly the strain-based compaction from Visage coupled run
- Hence, all compututations can be done as explicit coupling – no pore volume iterations were needed
- Extra effort: The tuning run (three explicit steps)
- BUT: We can get away with larger stress steps

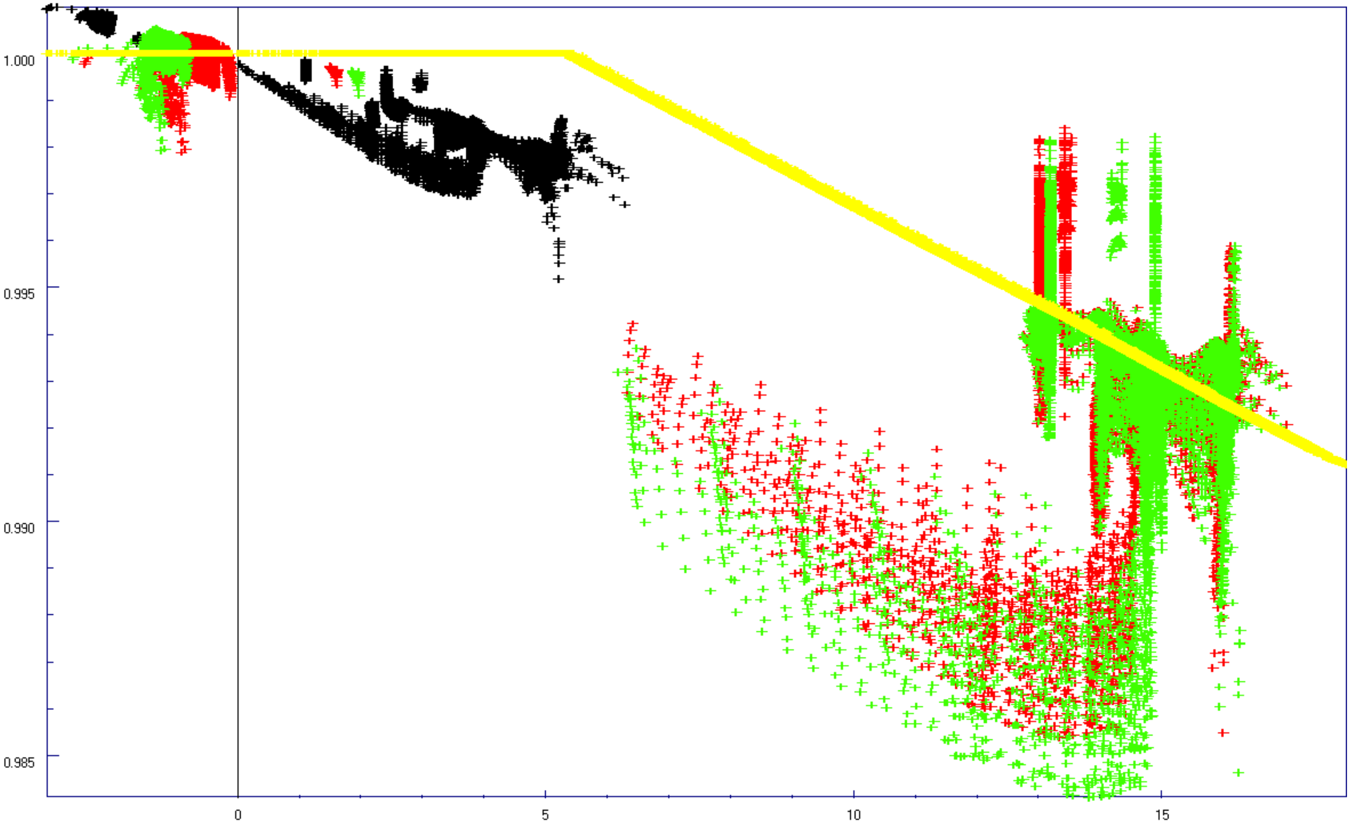
Multi-material chalk

Differences (from SM)
Analysis, discussion
Results, challenges

1. Construction of pseudo material regions
2. Construction of pseudo pvm-functions

Construction of pseudo Material Regions

Example: m_ε and m_{pf} from tuning run, three stress steps, base material 3 (transition zones)

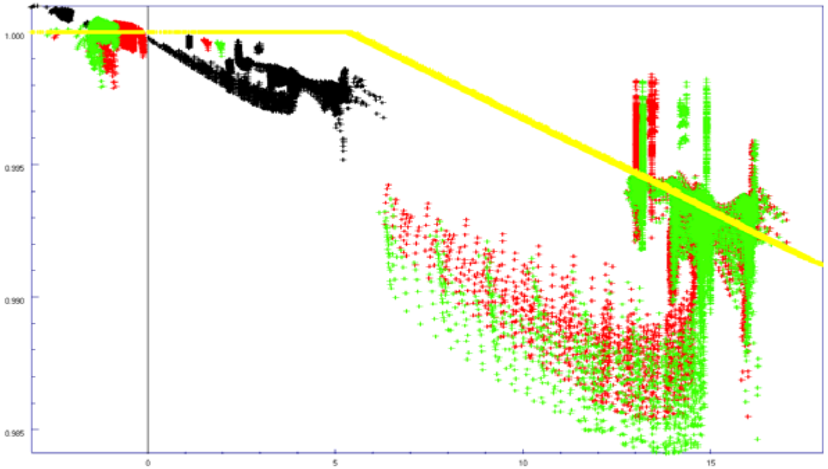


In contrast to the well-structured SM:

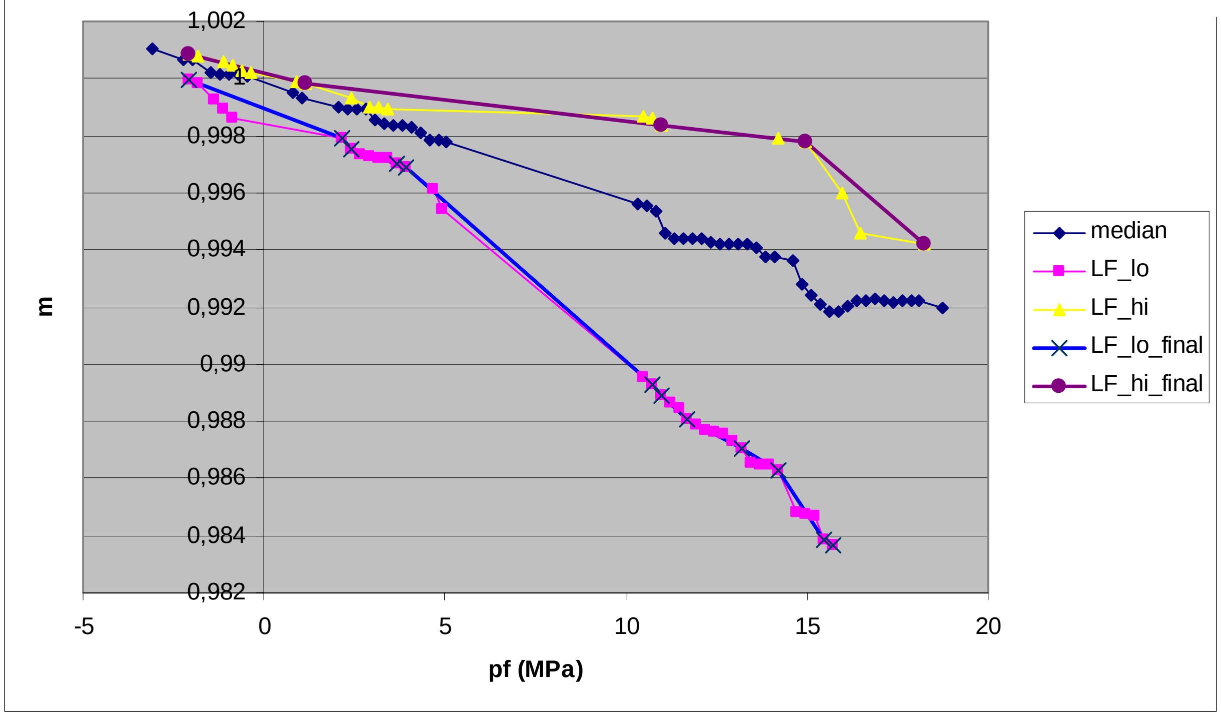
- Outliers
- False trends
- Handling of sparse data (Estimate missing trends)
- Non-Eclipse features (e.g. dilation)
- Desire for smoothness

Construction of pseudo Material Regions

Omitting the details (which are many) the envelope construction algorithm:



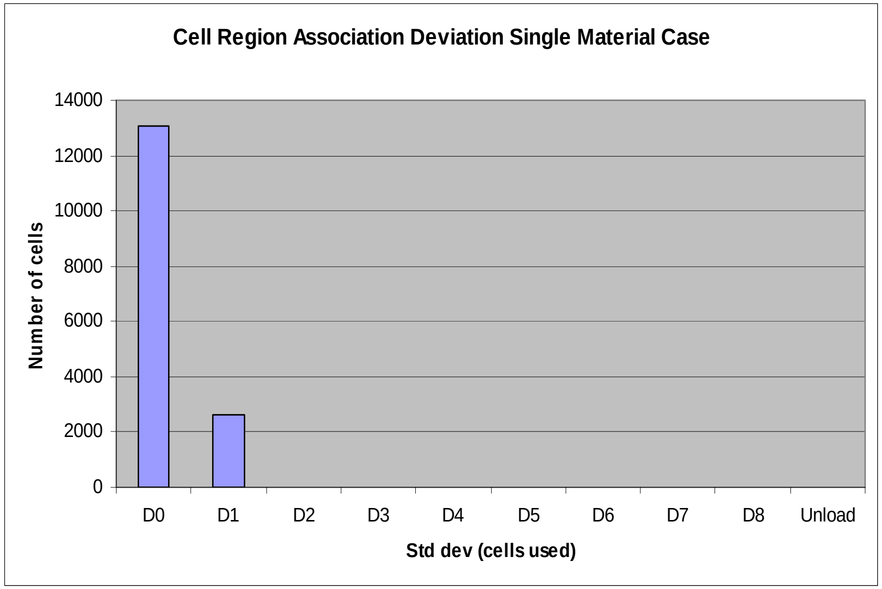
Levelfunctions regn 3, 3 stress steps

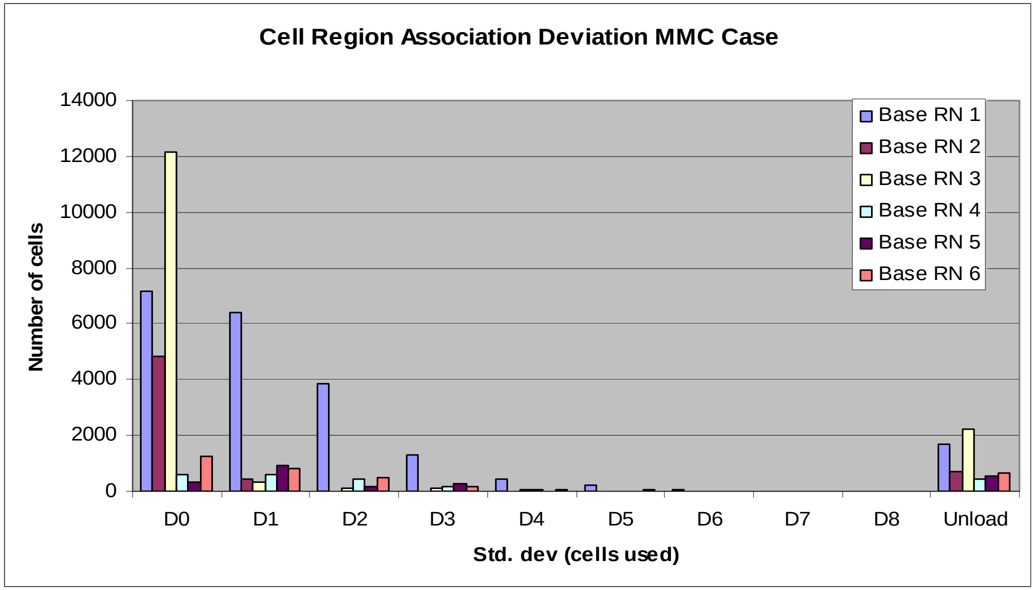


Construct median
Smoothen median
Remove outliers
Construct envelopes
Smoothen envelopes
Monotonize envelopes
Construct pseudo R_N (load-weighted, errors smallest for small loads)

Construction of pseudo Material Regions (2)

The ”almost static” hypothesis:
The pointcloud contains results from three different stress steps.
Ideally, any grid cell should be assigned to a unique
pseudo ROCKNUM irrespective of which stress step we regard.





For each grid cell, the ”standard deviation” of assigned ROCKNUM from different stress steps has been computed. The columns show #cells vs. stddev, SM left, MMC right (Unloading data are regarded as unsuitable for ROCKNUM assignment)

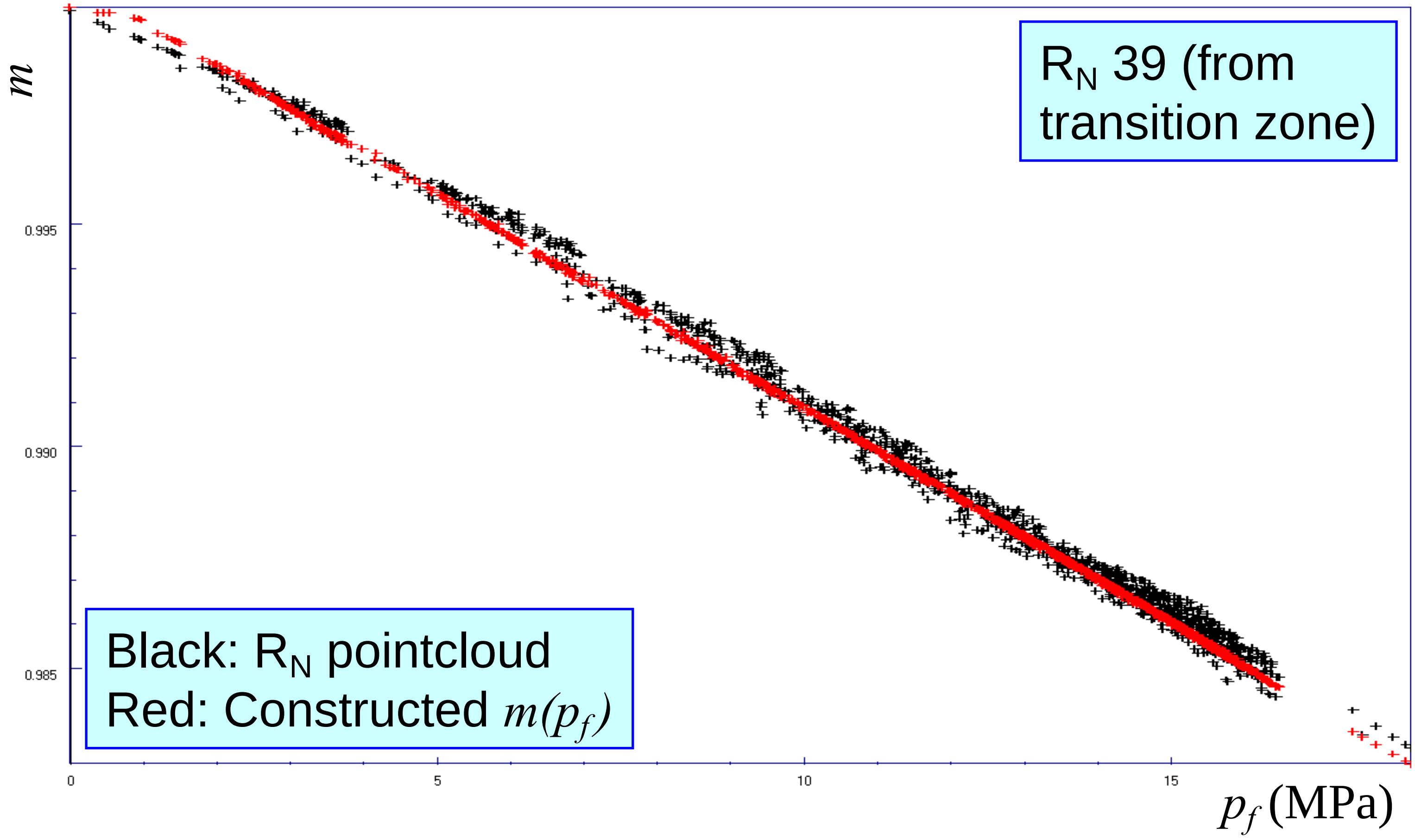
Construction of pseudo pvm-functions $m(p_f)$

Construction of $m(p_f)$ for a pseudo region R_N is based on analysis of the pointcloud for R_N, as defined in previous slides.

$m(p_f)$ must be Eclipse-acceptable, and should
- honour pointcloud trends
- be as smooth as possible without loss of information
- honour ”missing intervals” (trend estimate / guess)
- avoid false trends
- $m(0) = 1$ (no compaction at no load)

Observation: Incredibly easy to do by hand for ”any” pointcloud, but very difficult to develop a general and robust algorithm!

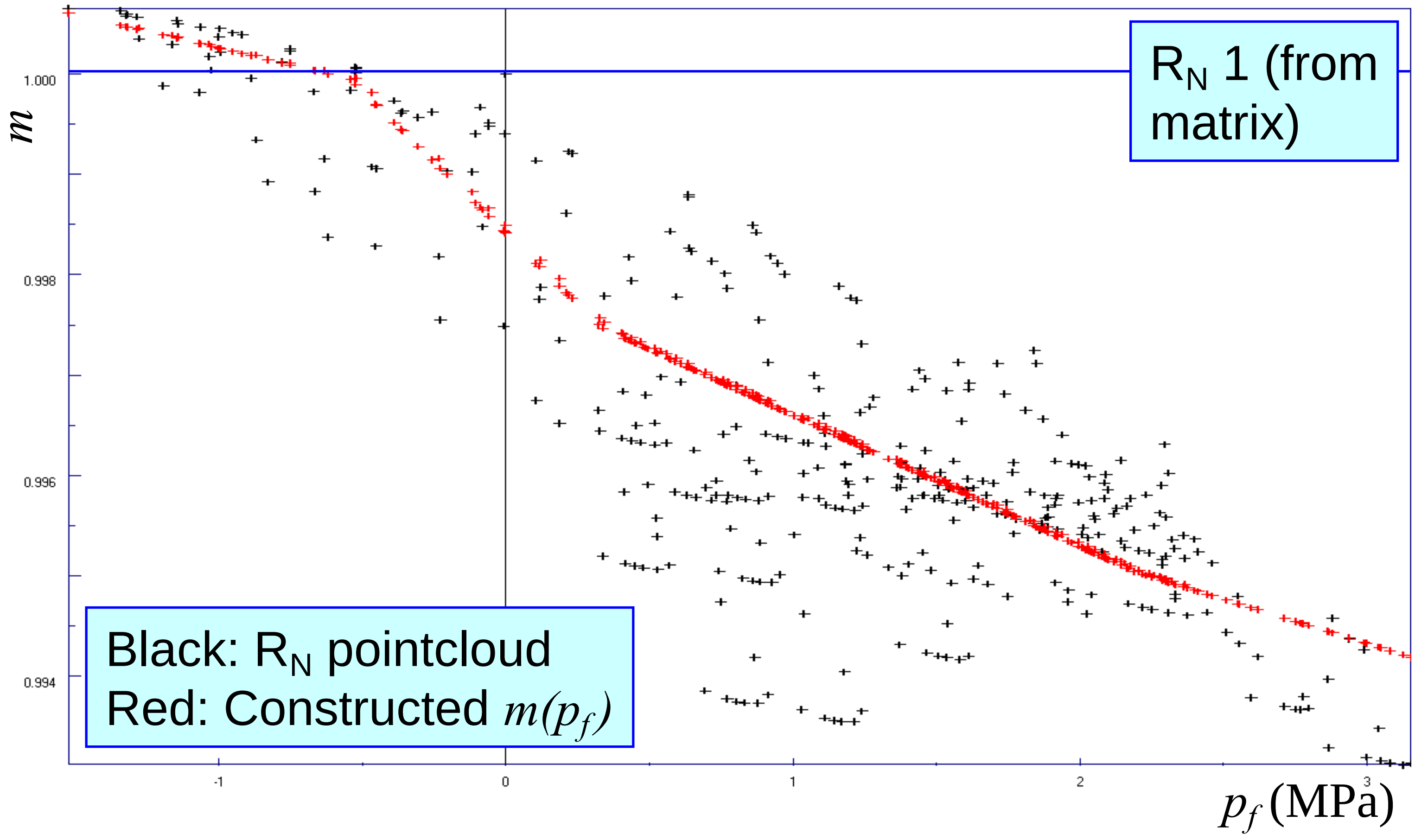
$m(p_f)$ construction: Straightforward case



Note 1: The R_N pointcloud is from final run w. 12 stress steps ($m(p_f)$ was constructed from three ssteps).

Note 2: A majority of the curves fall in this category!

$m(p_f)$ construction: Sparse pointcloud

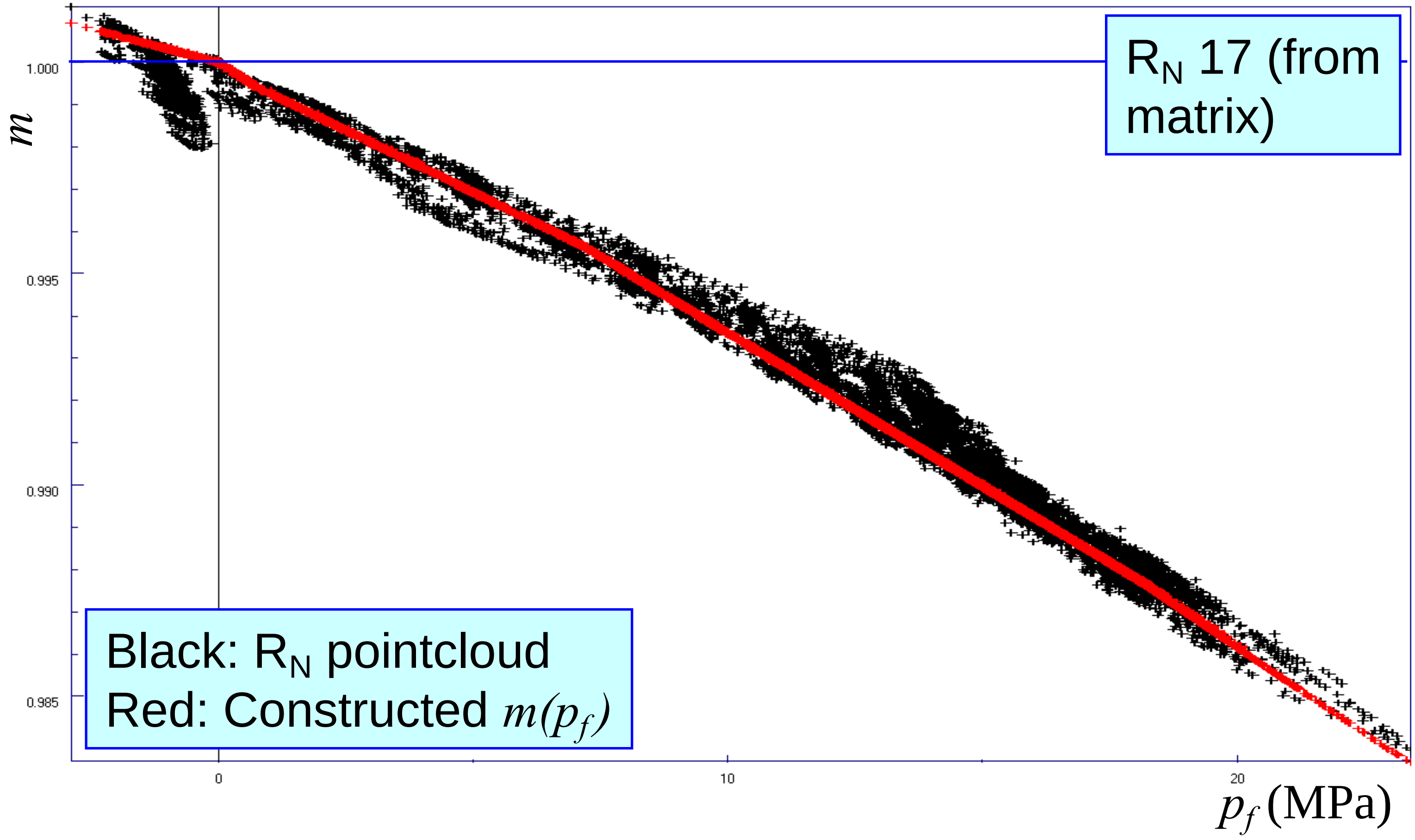


R_N 1 (from matrix)

Black: R_N pointcloud
Red: Constructed $m(p_f)$

Material boundary R_N, typically larger spread, does *not* pass through (0,1)

$m(p_f)$ construction: Confusing points near 0

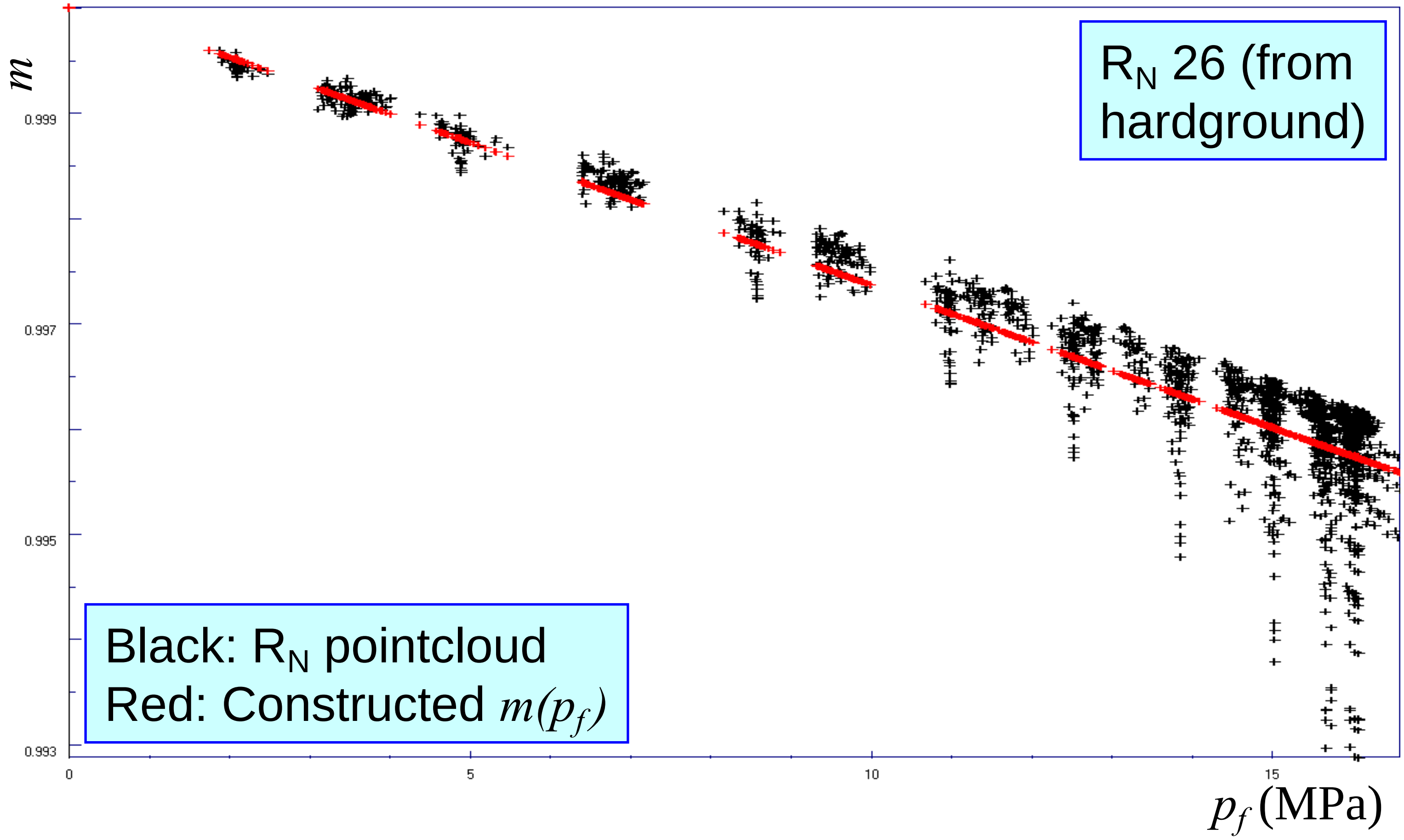


R$_N$ 17 (from matrix)

Black: R$_N$ pointcloud
Red: Constructed $m(p_f)$

Unloading part should be non-decreasing with unload
Honouring entire point-”blob” decreases quality of load part

$m(p_f)$ construction: Outliers

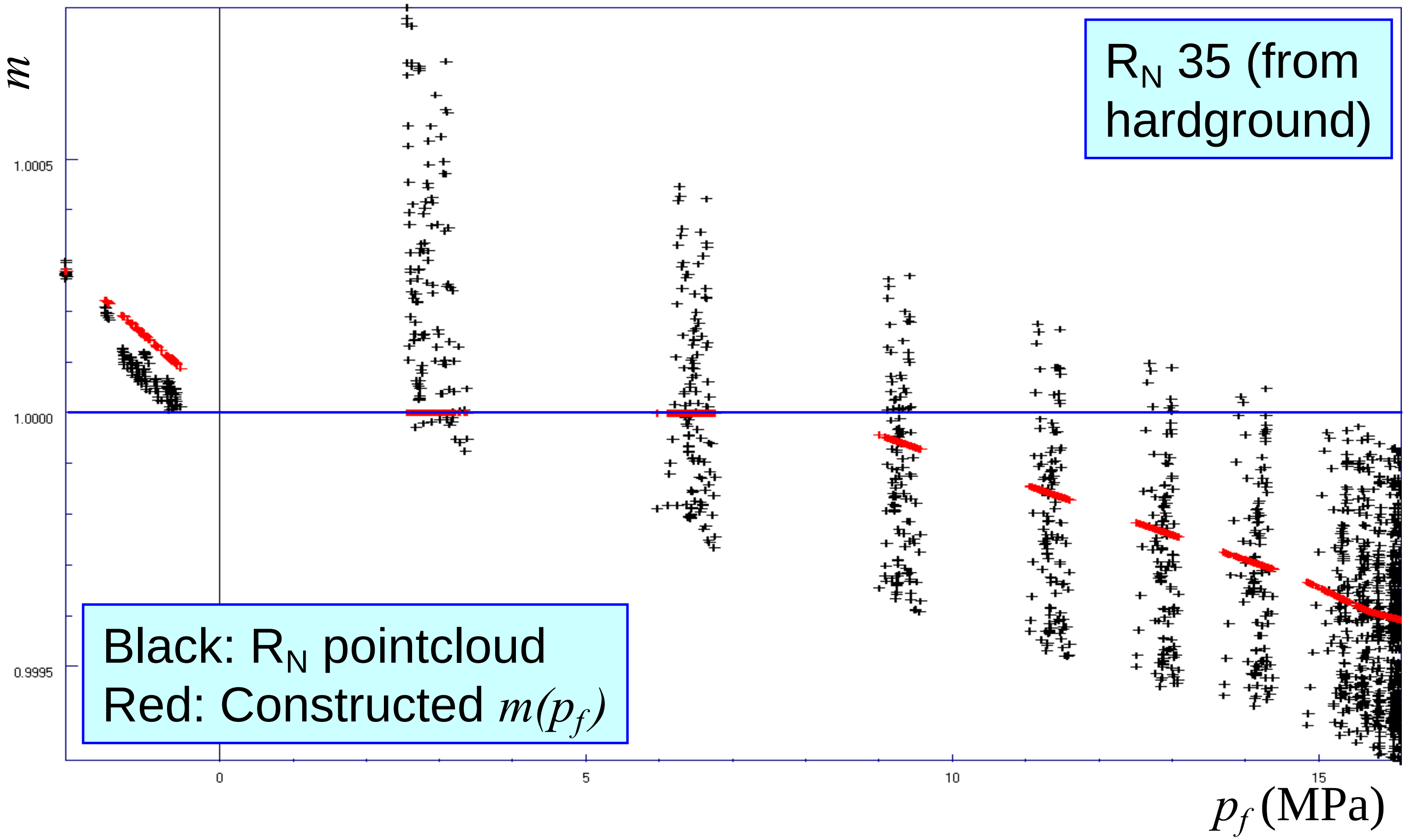


R$_N$ 26 (from hardground)

Black: R$_N$ pointcloud
Red: Constructed $m(p_f)$

Material boundary. Classify un-trendy points as outliers, which have been disregarded in construction

$m(p_f)$ construction: Dilation

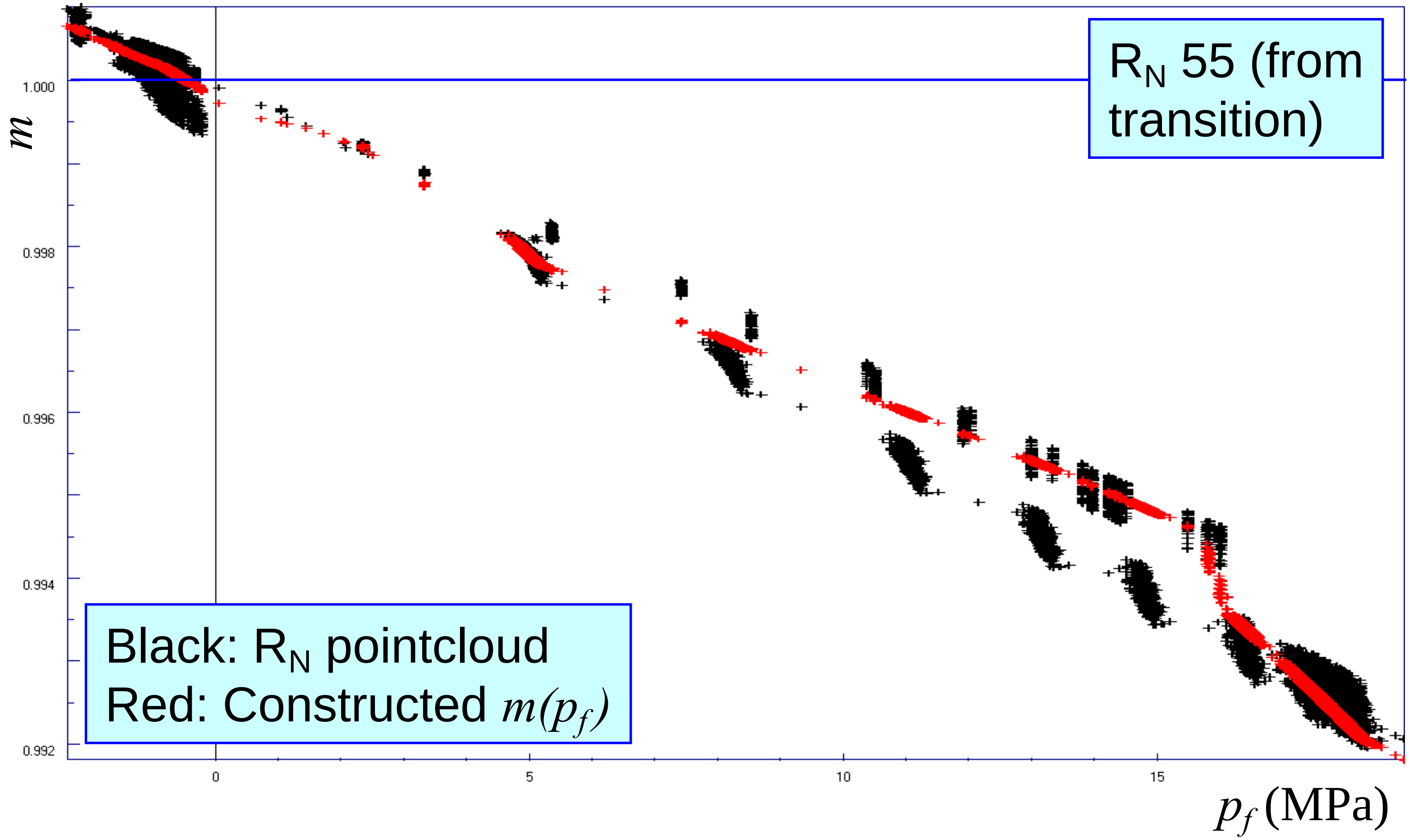


R_N 35 (from hardground)

Black: R_N pointcloud
Red: Constructed *$m(p_f)$*

Material boundary. Clearly dilation. Cannot be handled by Eclipse, and must be disregarded

$m(p_f)$ construction: Picking up the wrong trend (?)

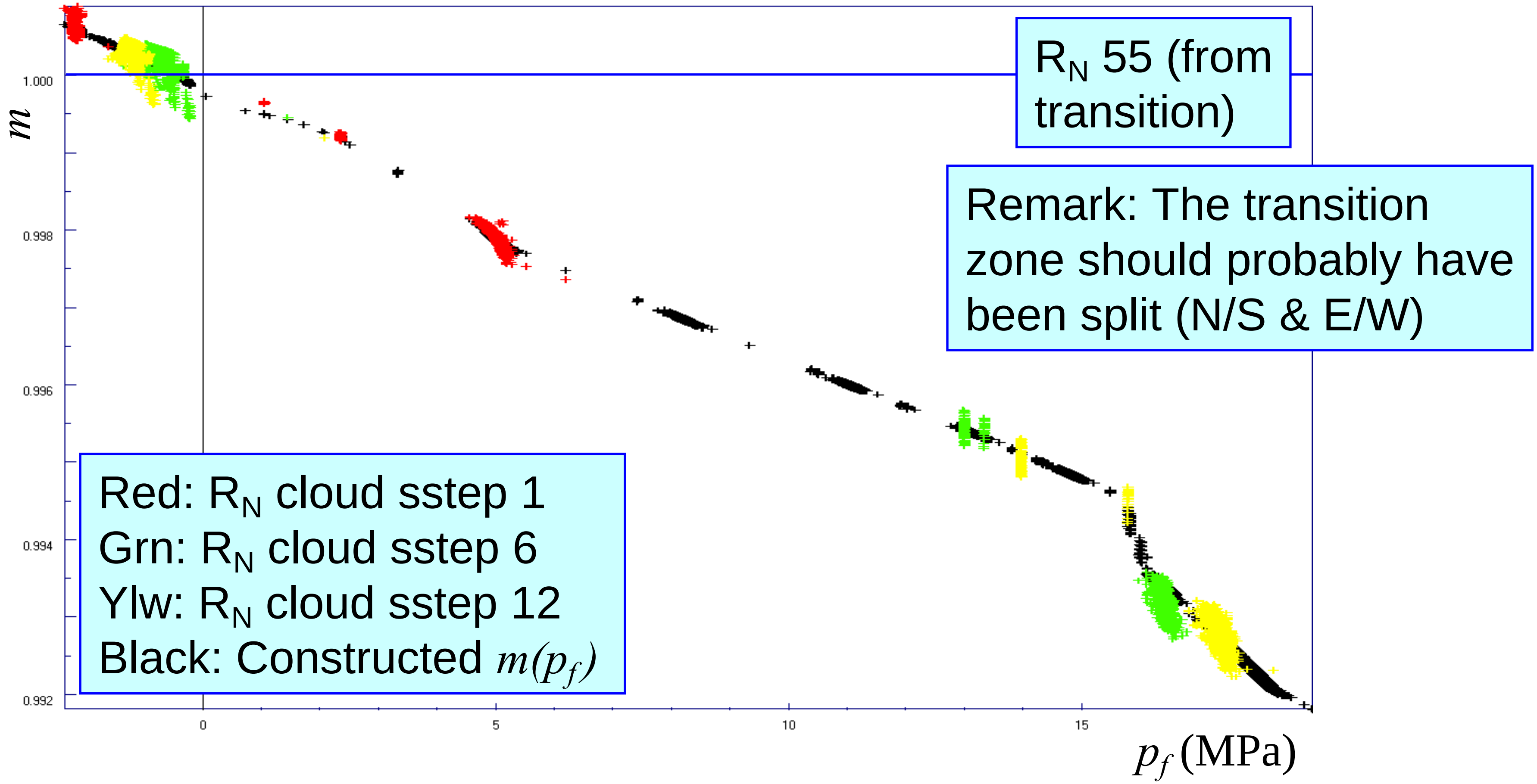


R_N 55 (from transition)

Black: R_N pointcloud
Red: Constructed $m(p_f)$

But...

$m(p_f)$ construction: Picking up the wrong trend (?)



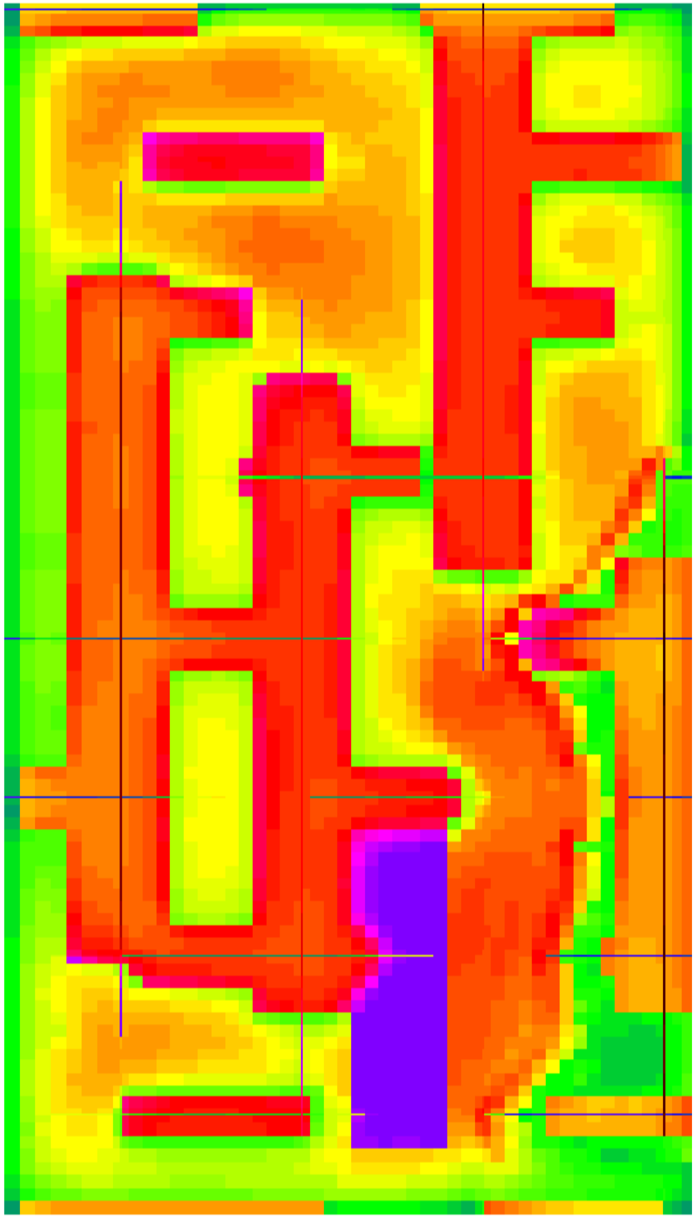
R_N 55 (from transition)

Remark: The transition zone should probably have been split (N/S & E/W)

Red: R_N cloud sstep 1
Grn: R_N cloud sstep 6
Ylw: R_N cloud sstep 12
Black: Constructed *$m(p_f)$*

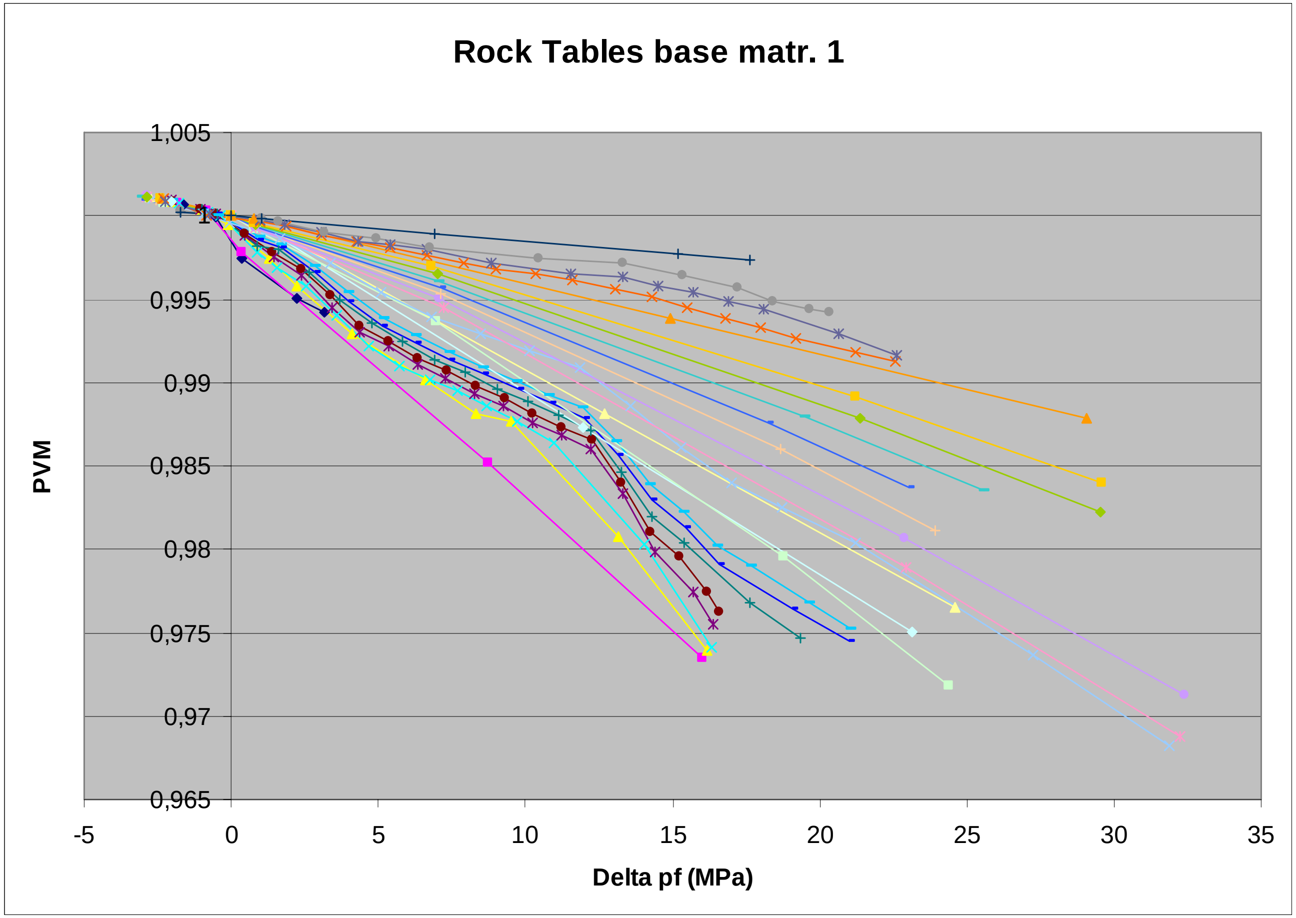
Showing the pointcloud that was actually produced by the tuning run reveals that this is as good as it can get

Pseudo material regions and tables

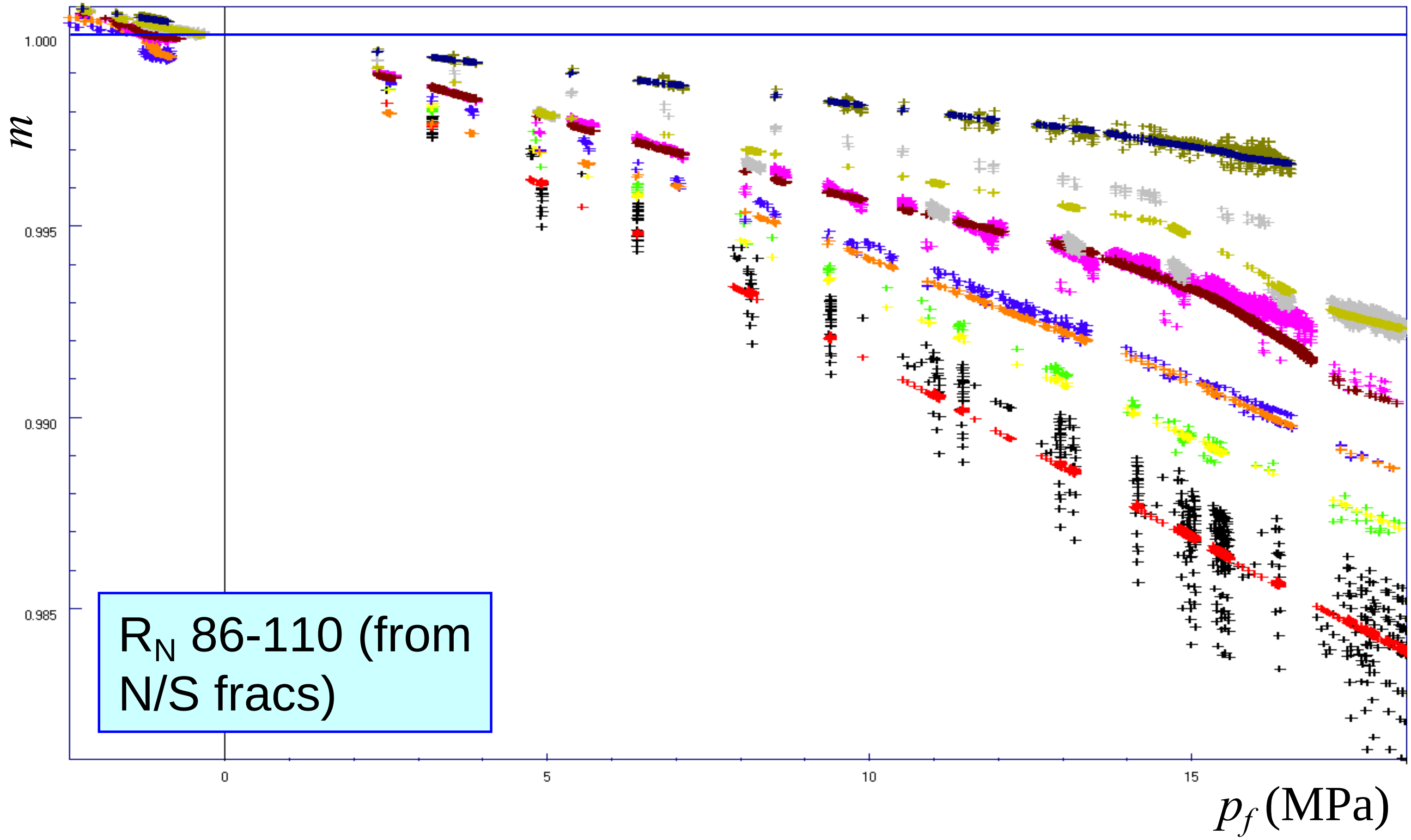


Regions reservoir layer 1 (left)

PVM-tables matrix (below)



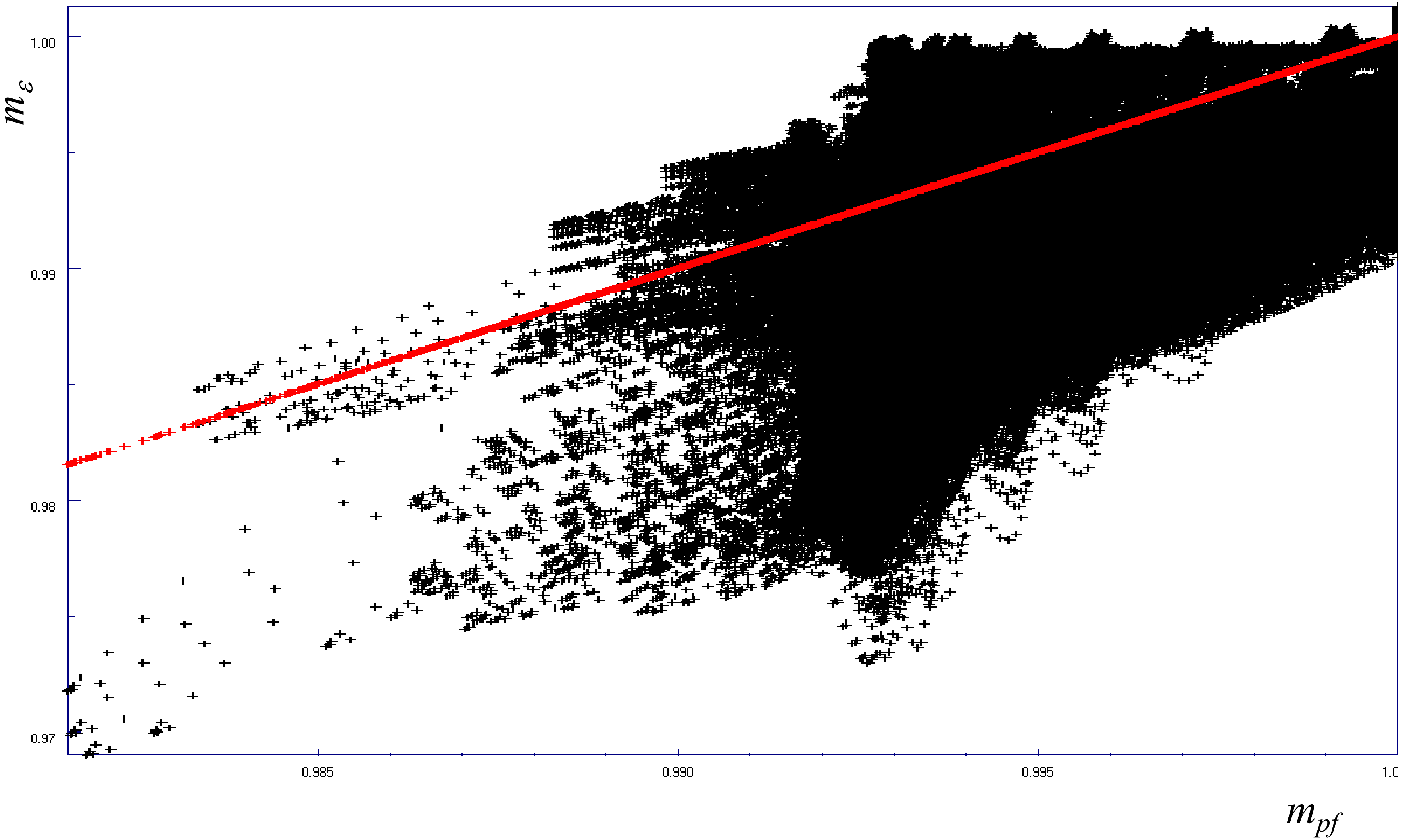
Example match: Some m_ε and m_{pf} from N/S fracs



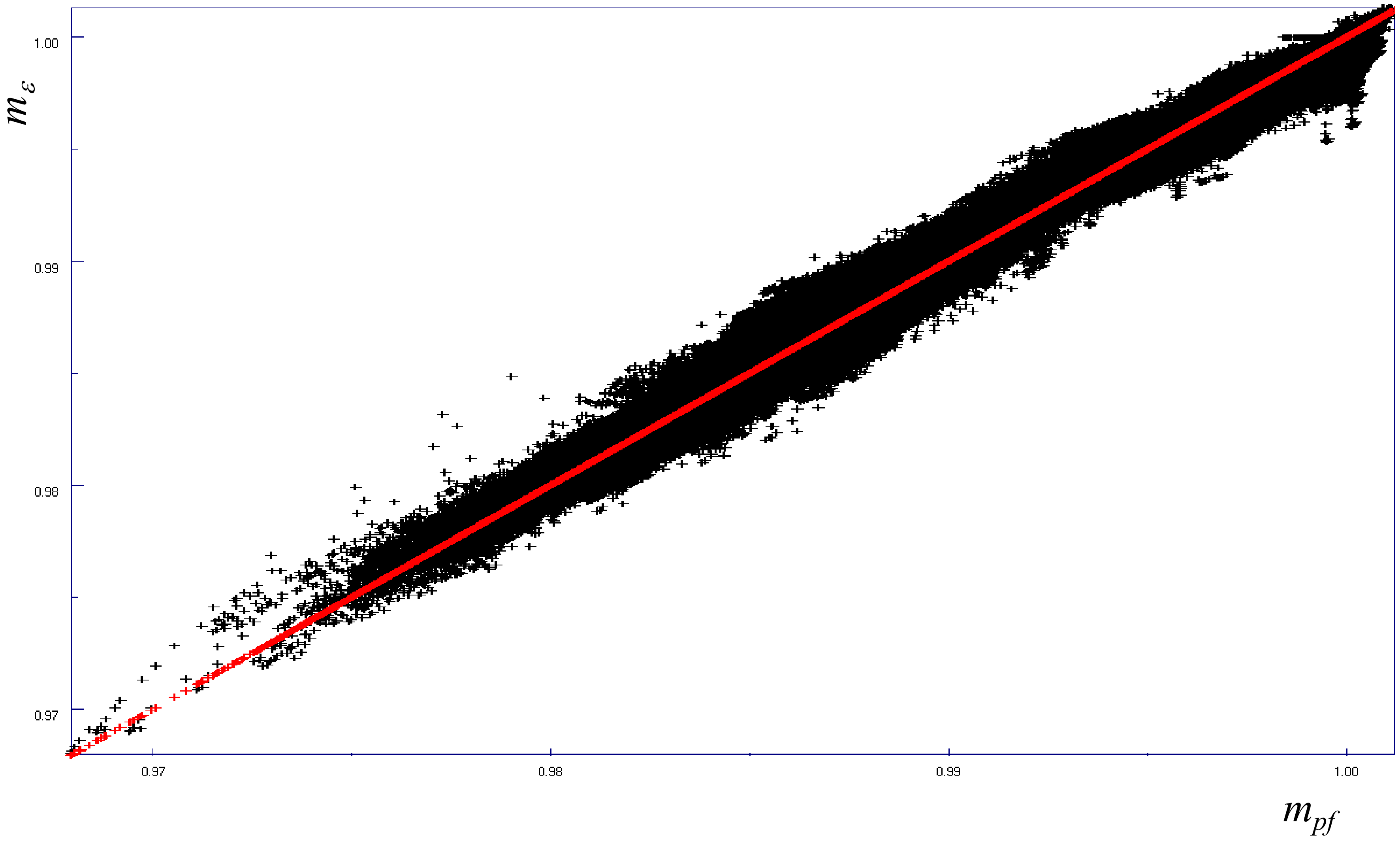
R_N 86-110 (from N/S fracs)

Note: Transforming back from load to pressure requires a careful estimate of no-load. (non-unique due to depth)

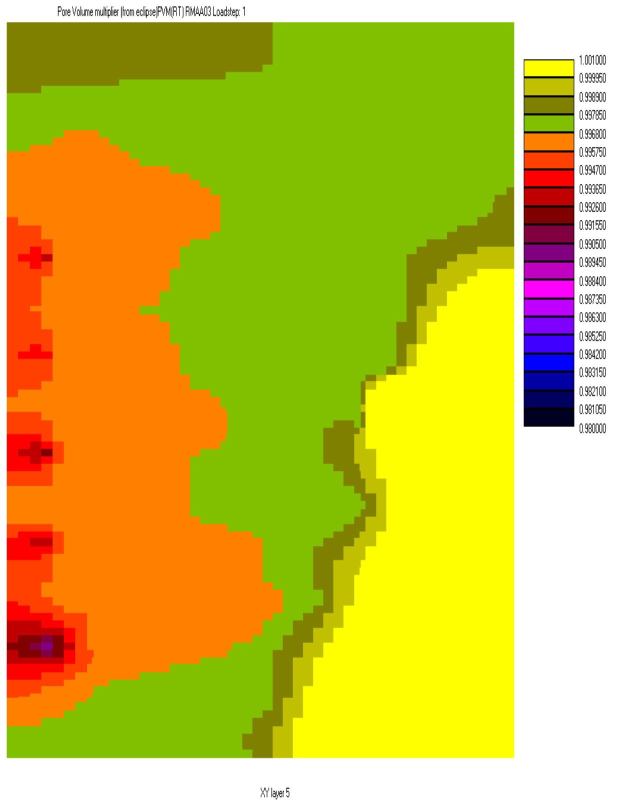
m_ε vs. m_{pf} base case ("optimal standard" Rock Tables)



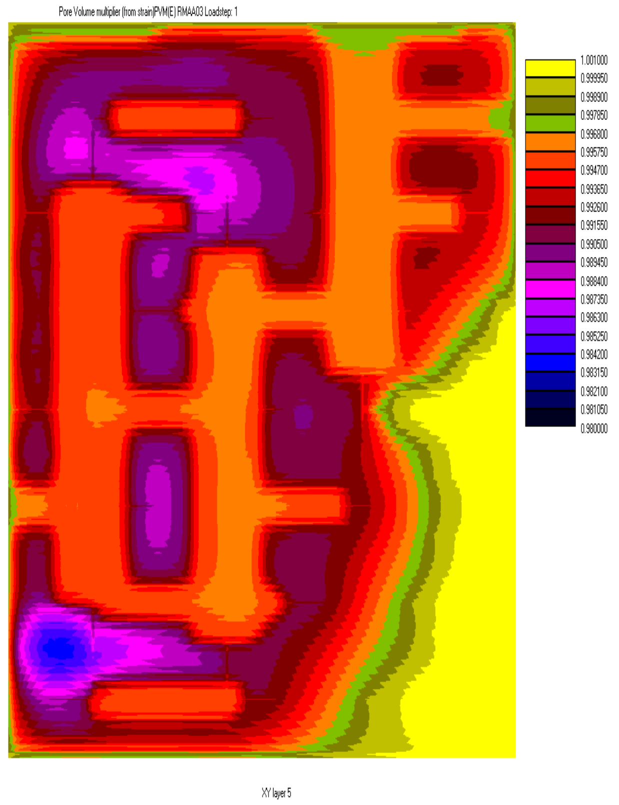
m_ε vs. m_{pf} with pseudo materials



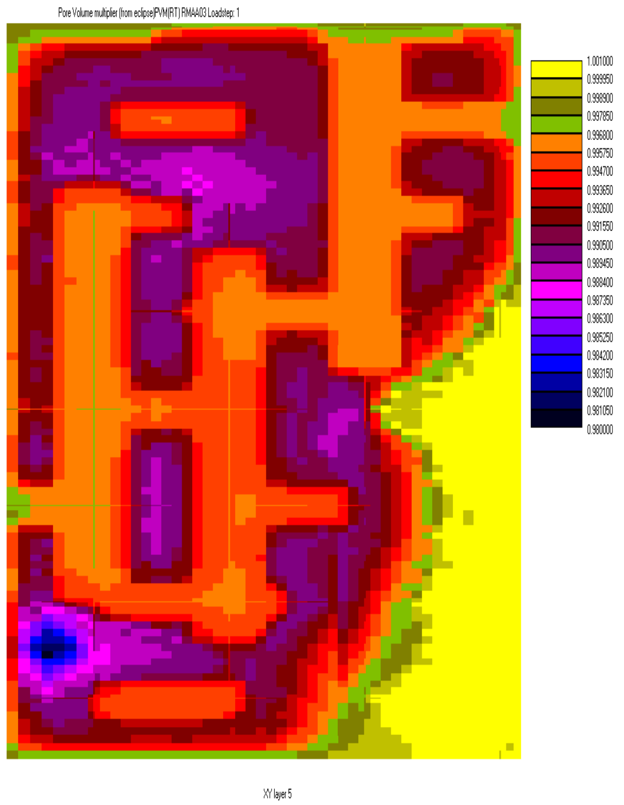
Contours of *m* (stress step 3, res. layer 5)



Base case,
from Eclipse
”classic” PVM-tables



Pseudo materials.
From Visage strain



Pseudo materials.
From Eclipse
PVM-tables

Note: Range is the same in all three figures

“Conservation of energy”

If flow-sim computed total compaction energy in all regions is correct, then $\sum_{F}^{n}$ is an optimal initialiser, and stress sim computed compaction energy will be correct.
(No PV-iterations needed)

The described construction procedure ensures that the compaction energy in all regions is accurate (disregarding outliers)

Some practical considerations (1)

The number of pseudo regions in a base region is determined by max. permitted pvm error at 10 MPa load. All parameters can be defaulted, or the user can specify the max error, min. and max number of subdivisions of any region, and % outliers to remove

Example input file:

```
# Max_error maxSlopeTV DiscardPercentage minNsub maxNsub
  0.0002        0.0          0.0            5       25
# Files
RMAA01.PUN  RMAA00.X0018
RMAA06.PUN  RMAA00.X0086
RMAA12.PUN  RMAA00.X0173
```

Some practical considerations (2)

Output from log file

```
Input files for model: 'RMAA'
Initial files (orgData, Data, init, pun, x)
  RMAA.DATA RMAA00.DATA RMAA00.INIT RMAA00.X0000 RMAA00.PUN
Dynamic files (pun, x)
  RMAA01.PUN  RMAA00.X0018
  RMAA06.PUN  RMAA00.X0086
  RMAA12.PUN  RMAA00.X0173
Total number of (Eclipse) cells read: 127920, active cells: 50025
Loading dynamic files RMAA01.PUN & RMAA00.X0018 :
Loading dynamic files RMAA06.PUN & RMAA00.X0086 :
Loading dynamic files RMAA12.PUN & RMAA00.X0173 :

**********************************************************************
**                                                                  **
** Org. RockNum  1 subdivided into 25 materials, ROCKNUM   1 to  25 **
** Org. RockNum  2 subdivided into 10 materials, ROCKNUM  26 to  35 **
** Org. RockNum  3 subdivided into 25 materials, ROCKNUM  36 to  60 **
** Org. RockNum  4 subdivided into 25 materials, ROCKNUM  61 to  85 **
** Org. RockNum  5 subdivided into 25 materials, ROCKNUM  86 to 110 **
** Org. RockNum  6 subdivided into 25 materials, ROCKNUM 111 to 135 **
**                                                                  **
** Max number of entries in a table:  23                            **
**                                                                  **
**********************************************************************
```

Computing time analysis

Using pseudo materials, Eclipse will compute
very accurate pressures and pore volume multipliers
(as compared to the ”correct” Visage values)

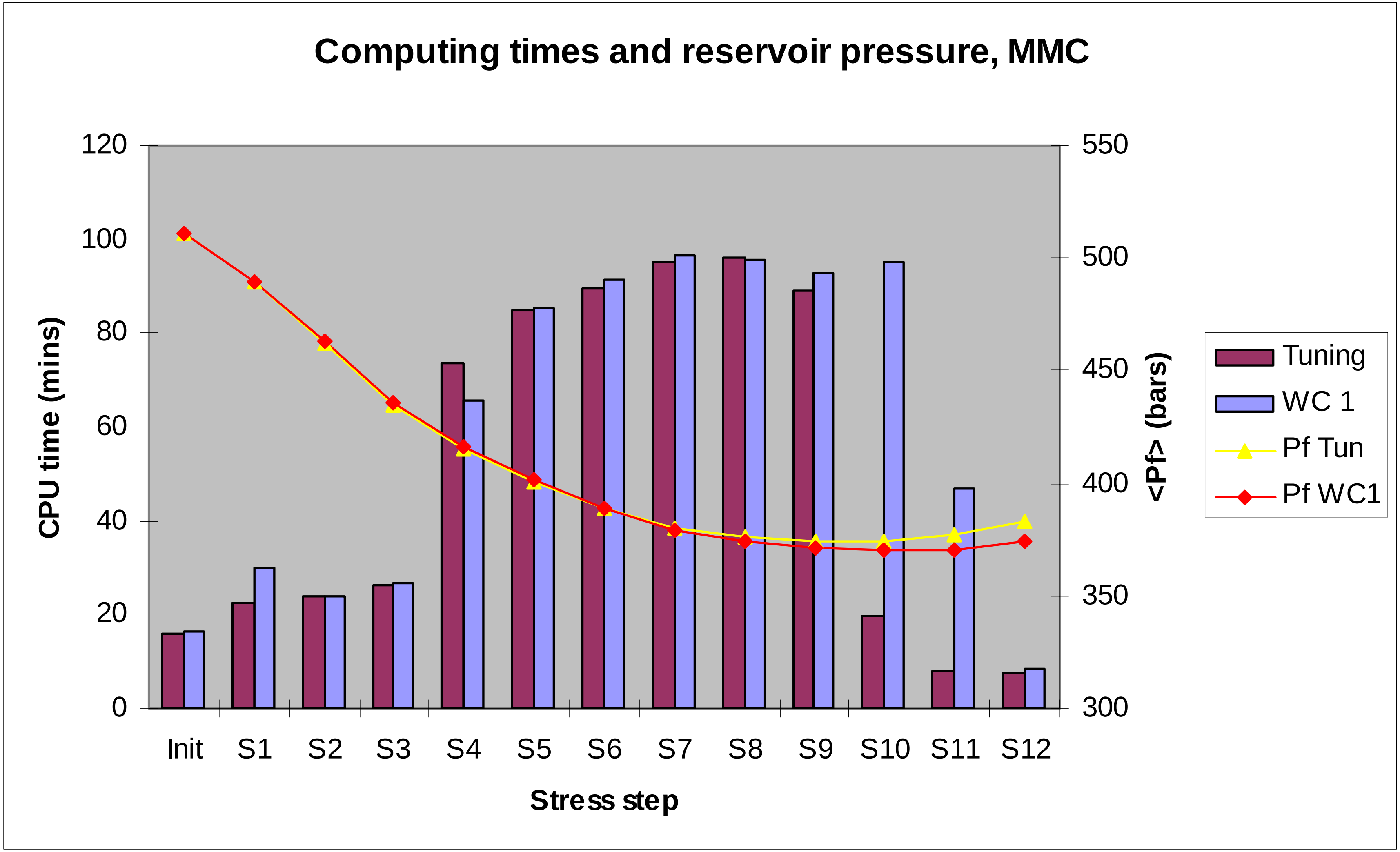
The reservoir state Σ_F is therefore a very good initialiser
for both the solver iterations and pore volume iterations.

By the results above, pore volume iterations will normally
not be needed, or at least significantly fewer necessary.

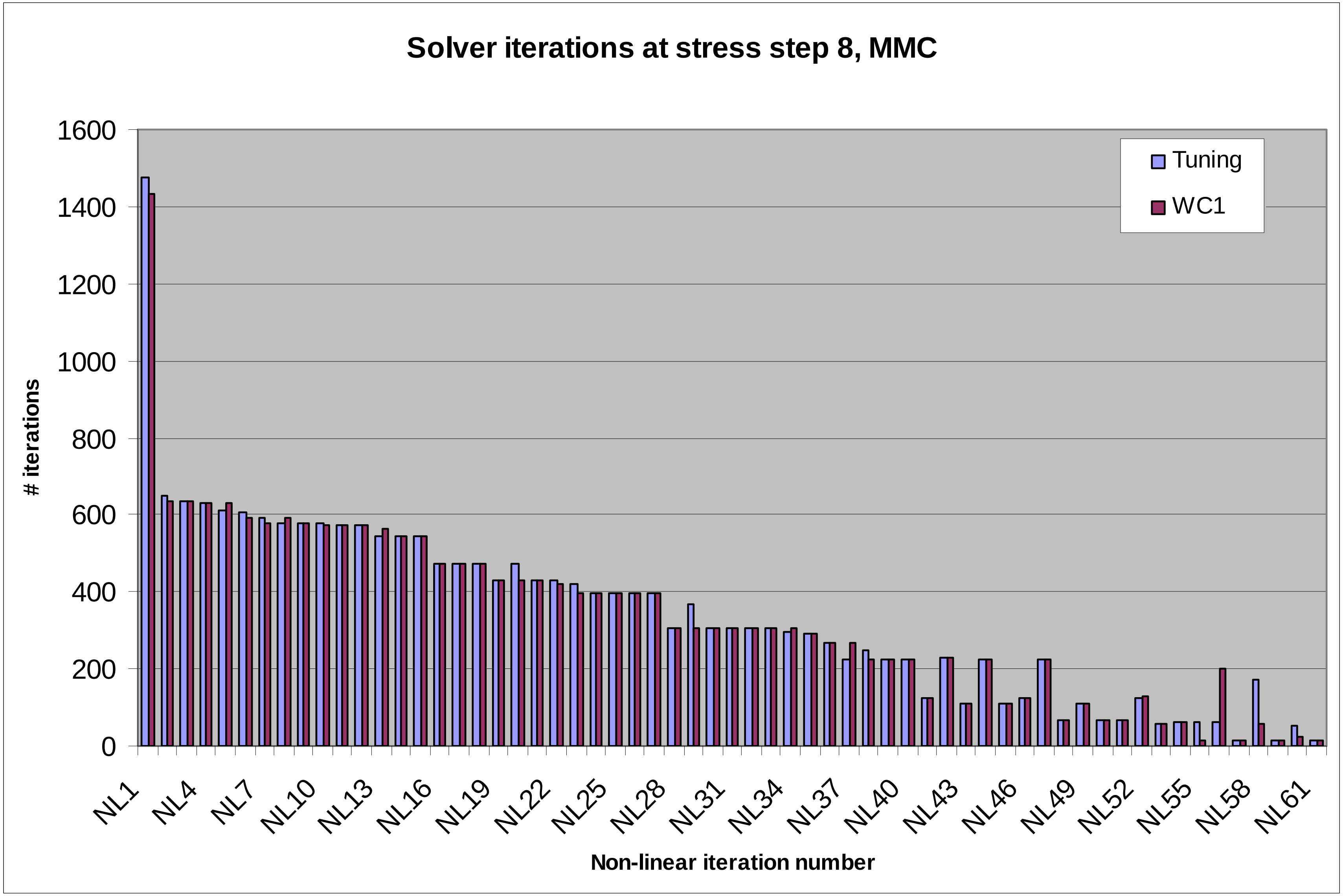
We would also expect fewer solver iterations.

But...

CPU times base MMC and with pseudo matrs



Outer and inner iterations Visage



Computing time analysis (2)

CPU-times and especially iteration behaviour are almost identical in the two runs, and seem to be completely independent of the flow state initialiser.

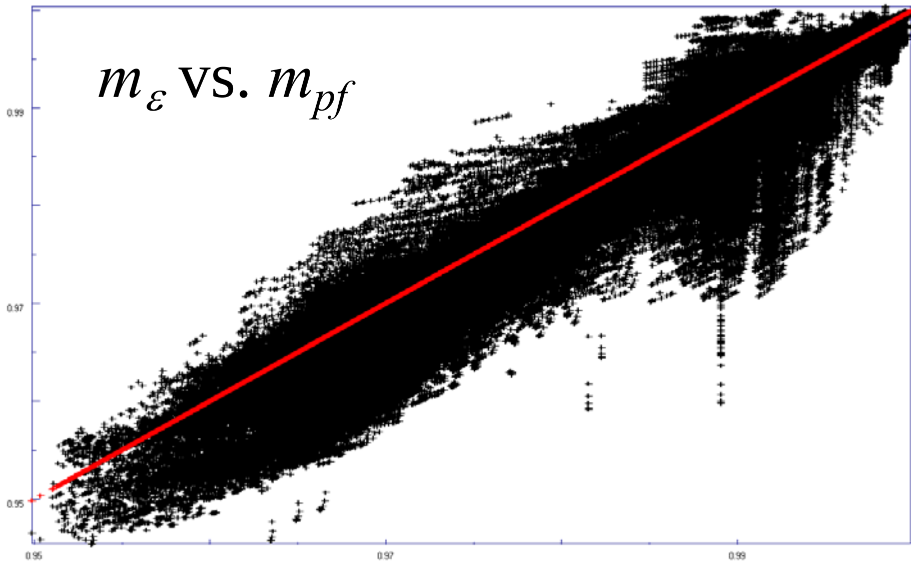
Recall,

$$\Sigma_{init}^{n} = (\Sigma_{R}^{n-1}, \Sigma_{F}^{n})$$

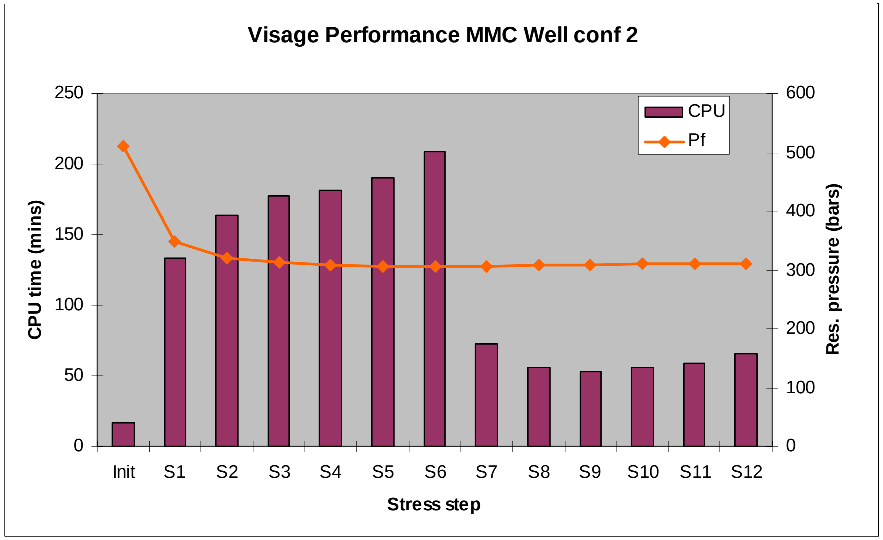
For the solver iterations, Σ_{R}^{n-1} is the dominating factor, while Σ_{F}^{n} is most important for the pore volume iterations

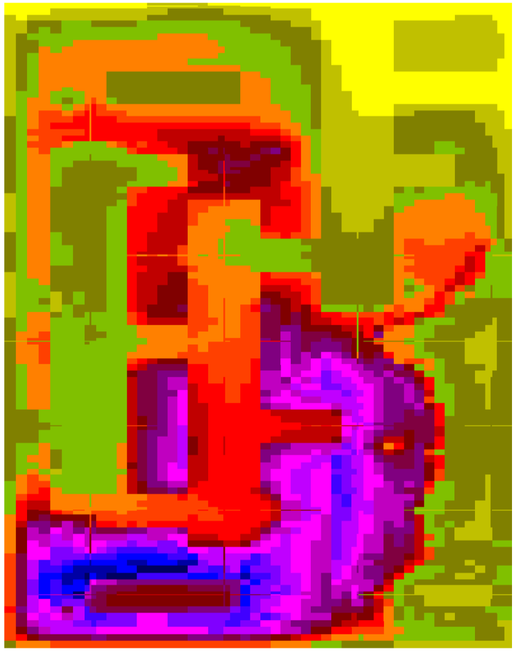
I.e. We cannot speed up the Visage solver at a single stress step by improving the flow state.

Sensitivity: “Random” Well pattern & rates, same pvm-tables as before

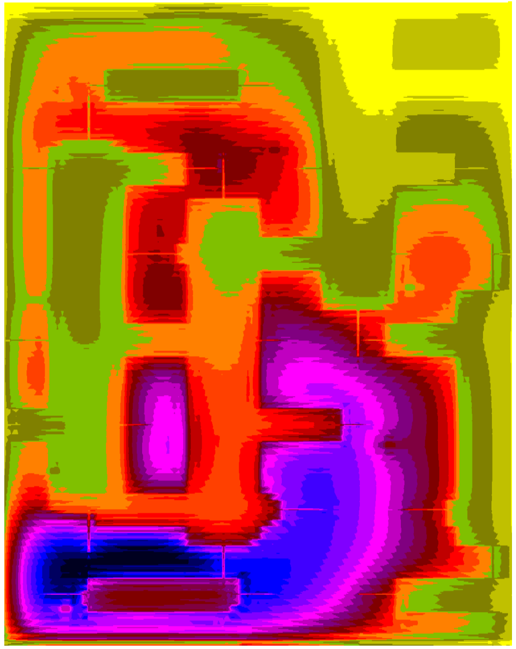


m_{ε} vs. m_{pf}





m_{pf}, res. layer 2, stress step 2



m_{ε}, res. layer 2, stress step 2

Conclusions

By a small extra effort (”tuning run”), compaction description in the flow simulator can be vastly improved such that the reservoir state computed by it (almost) perfectly matches the ”correct” state as delivered from the stress simulator.

Pore volume iterations are eliminated or reduced

The reservoir state is ”correct” at all times, not only at stress steps.

Better control of reservoir state allows for larger stress steps

The procedure seems to be robust and reliable.