Savannah River National Laboratory™

OPERATED BY SAVANNAH RIVER NUCLEAR SOLUTIONS

We put science to work.™

Best Practices / Lessons Learned from Software Qualification and Model Verification

Greg Flach
Environmental Modeling

Interagency Steering Committee on Performance and Risk Assessment Community of Practice Annual Technical Exchange Meeting

October 19-20, 2016

SRNL-STI-2016-00615

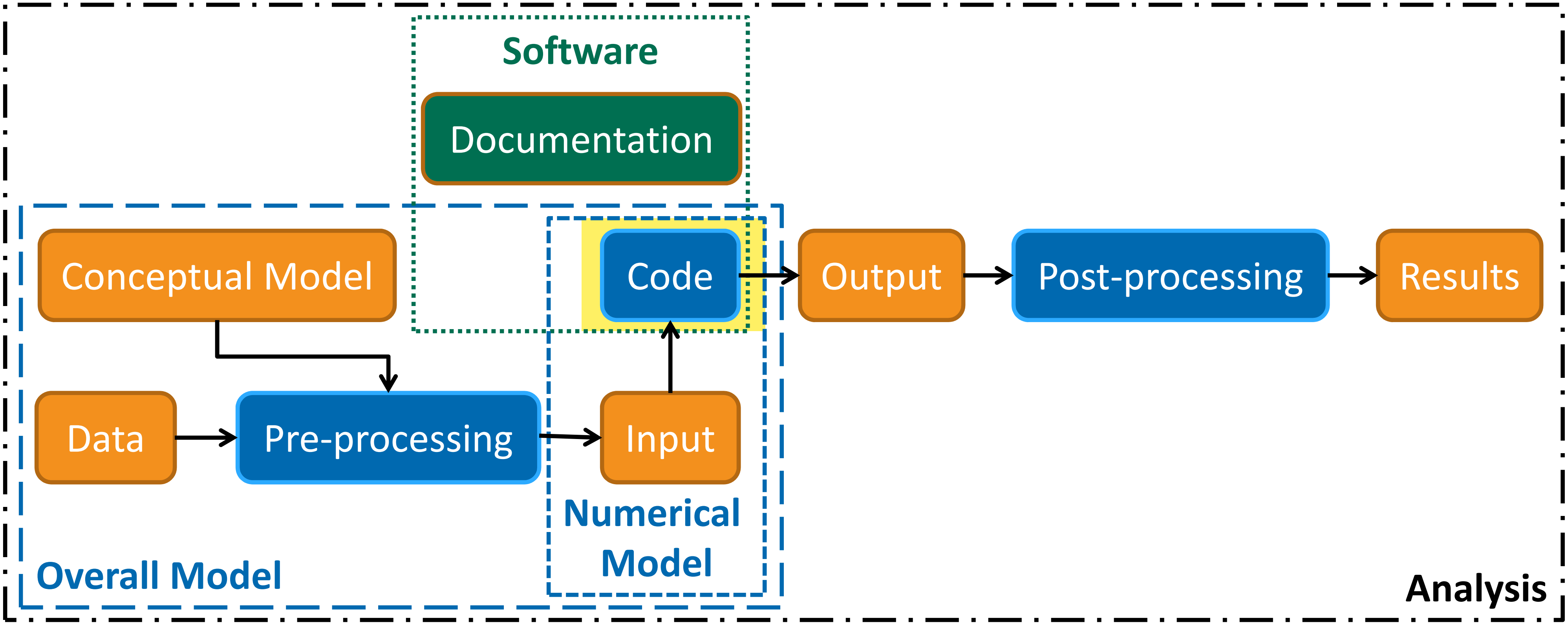
Savannah River NUCLEAR SOLUTIONS℠

FLUOR • NEWPORT NEWS NUCLEAR • HONEYWELL

Outline

- Models and Analyses are more than Codes
- Quality Assurance (QA) versus Quality Control (QC) approaches
- QA / QC Risks and Remedies
- Examples
- Summary thoughts
- Audience participation

Components of a Risk or Performance Assessment Analysis Involving Simulation



Software, Models and Analyses encompass more than Codes:

- Software = Code + Documentation
- Numerical Model = Input + Code
- Overall Model = Concept + Data + Input + Code
- Analysis = All of the above

Quality Assurance must address the entire Analysis to support sound Decisions

Quality Assurance (QA) versus Quality Control (QC) Paradigms

In my experience:

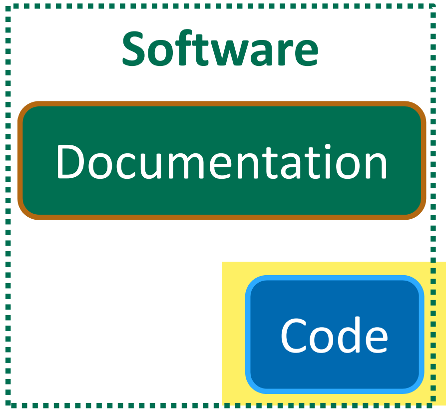
- QA and QC are complementary
- QA takes forefront w.r.t software
- QC takes forefront w.r.t model or analysis

Many people still use the term Quality Assurance (QA) and Quality Control (QC) interchangeably but this should be discouraged.

Criteria	**Software Quality Assurance (SQA)**	**Software Quality Control (SQC)**
Definition	SQA is a set of activities for ensuring quality in software engineering processes (that ultimately result in quality in software products). The activities establish and evaluate the processes that produce products.	SQC is a set of activities for ensuring quality in software products. The activities focus on identifying defects in the actual products produced.
Focus	Process focused	Product focused
Orientation	Prevention oriented	Detection oriented
Breadth	Organization wide	Product/project specific
Scope	Relates to all products that will ever be created by a process	Relates to specific product
Activities	• Process Definition and Implementation • Audits • Training	• Reviews • Testing

http://softwaretestingfundamentals.com/sqa-vs-sqc/

Software Quality Assurance / Qualification



Best Practices

- "QA" approach: Check Once, Use Many
  - Simulation software embodies relatively general and static functions
  - Application of systematic software QA activities through formal procedures worth the investment

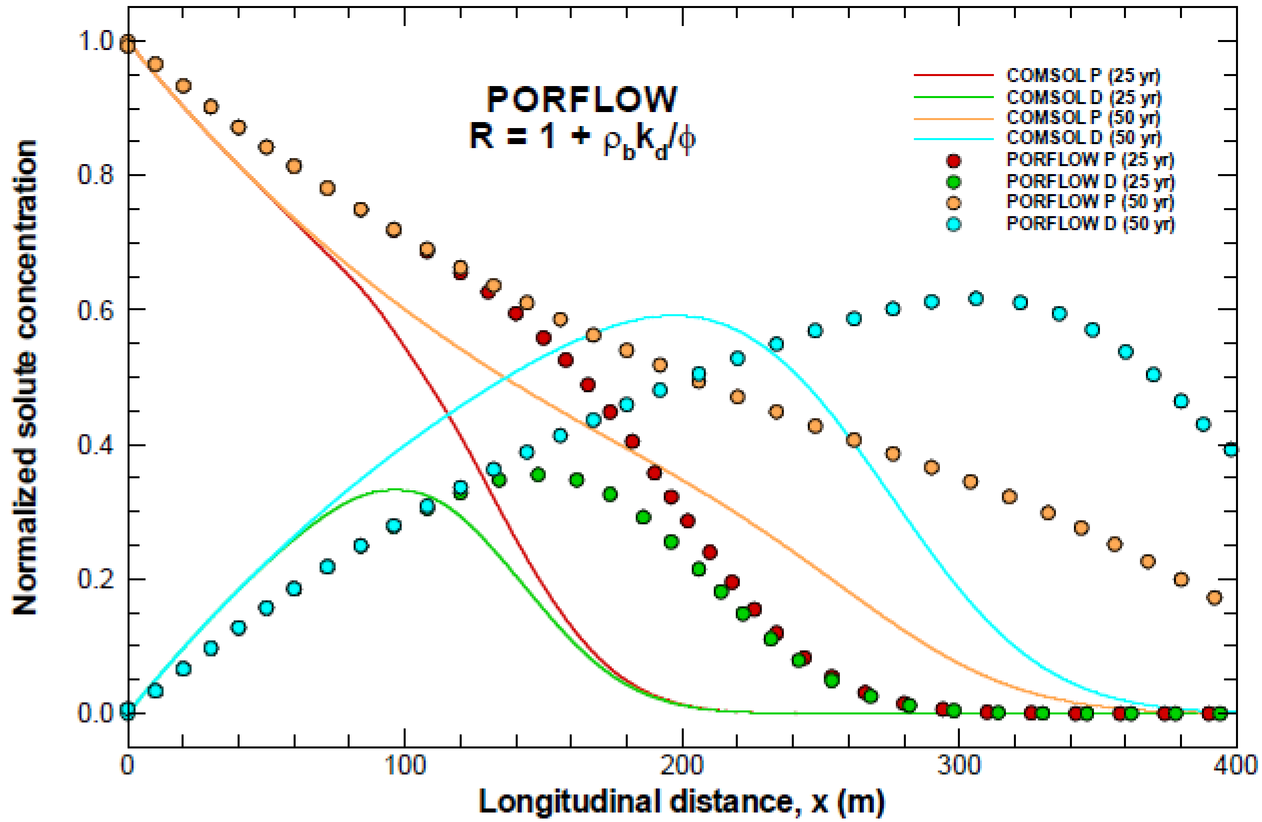
Experience / Lessons Learned

- Input errors and misuse more common than coding errors
  - Often due to code functionality that is ambiguously, obscurely or not documented
  - Documentation, not code, problem
- The more code use the better (early use, large user base)
- Developer-only testing is often lacking

Example: Retardation Coefficient, R

Conventional definition of R

$$R = 1 + \frac{\rho_s(1-n)K_d}{Sn}$$



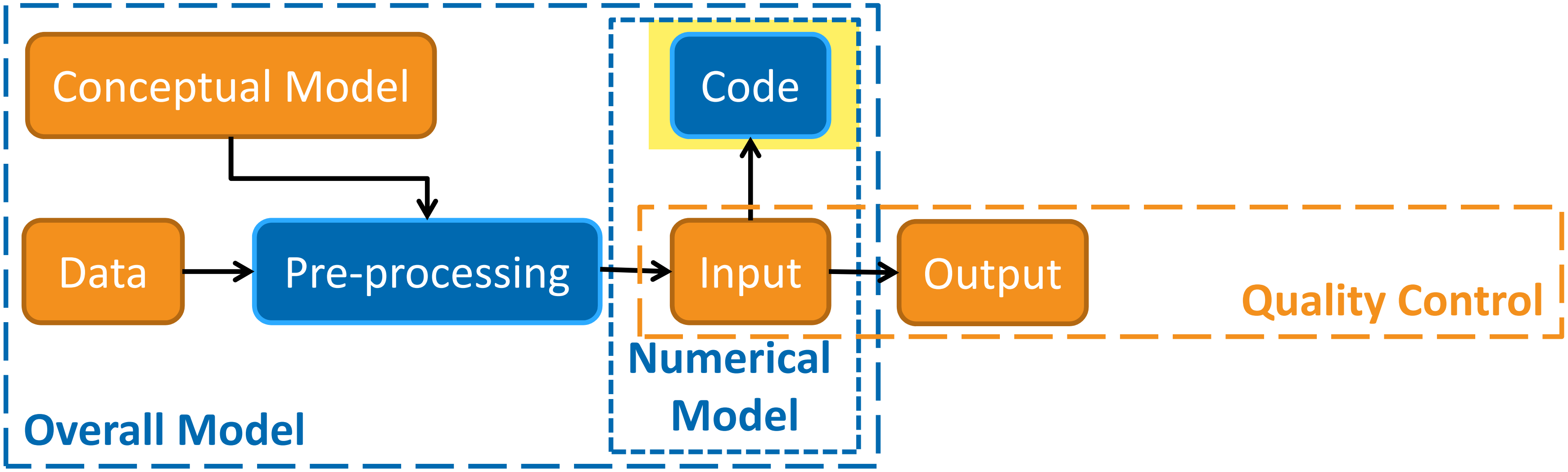
Unusual alternative definition <u>and Porflow default</u>

$$R = 1 + \frac{\rho_s(1-Sn)K_d}{Sn}$$

Diagnosis:

- QA testing had involved only fully saturated cases (S = 1)
- Discovered through code-to-code benchmarking during model abstraction
- User not alerted to non-conventional retardation definition in documentation

Model Quality Assurance / Verification



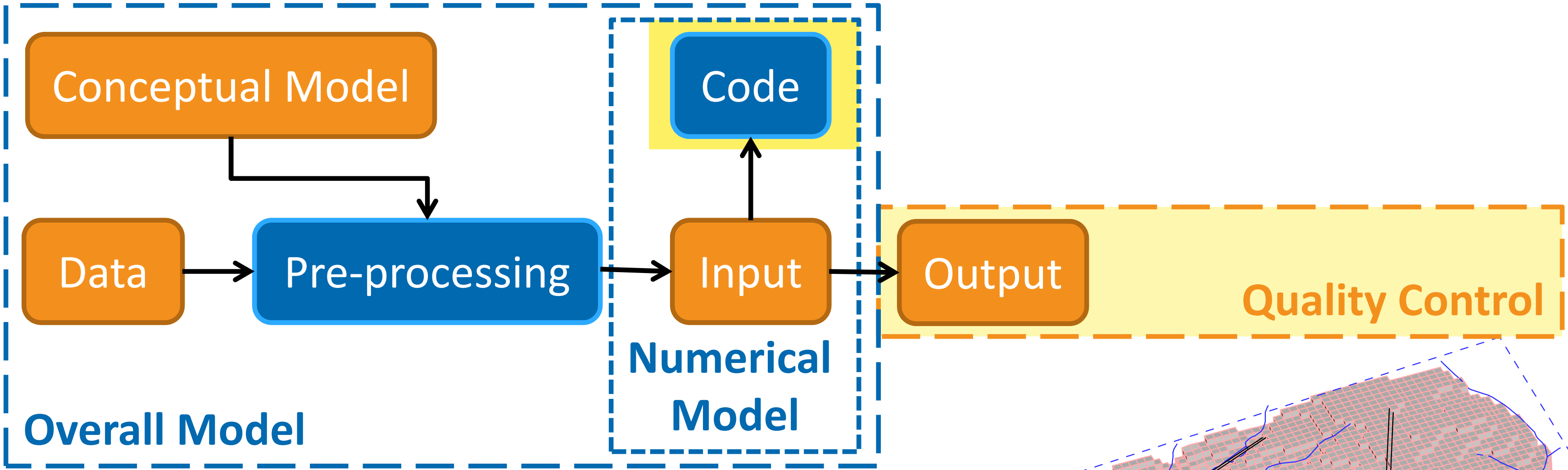
Best Practices

- "QC" approach: Check Every Use (in addition to QA for Code)
  - Pre-processing software is embodies specific and undocumented functions
  - Input and/or Output checked every time is more efficient than formal software QA

Experience / Lessons Learned

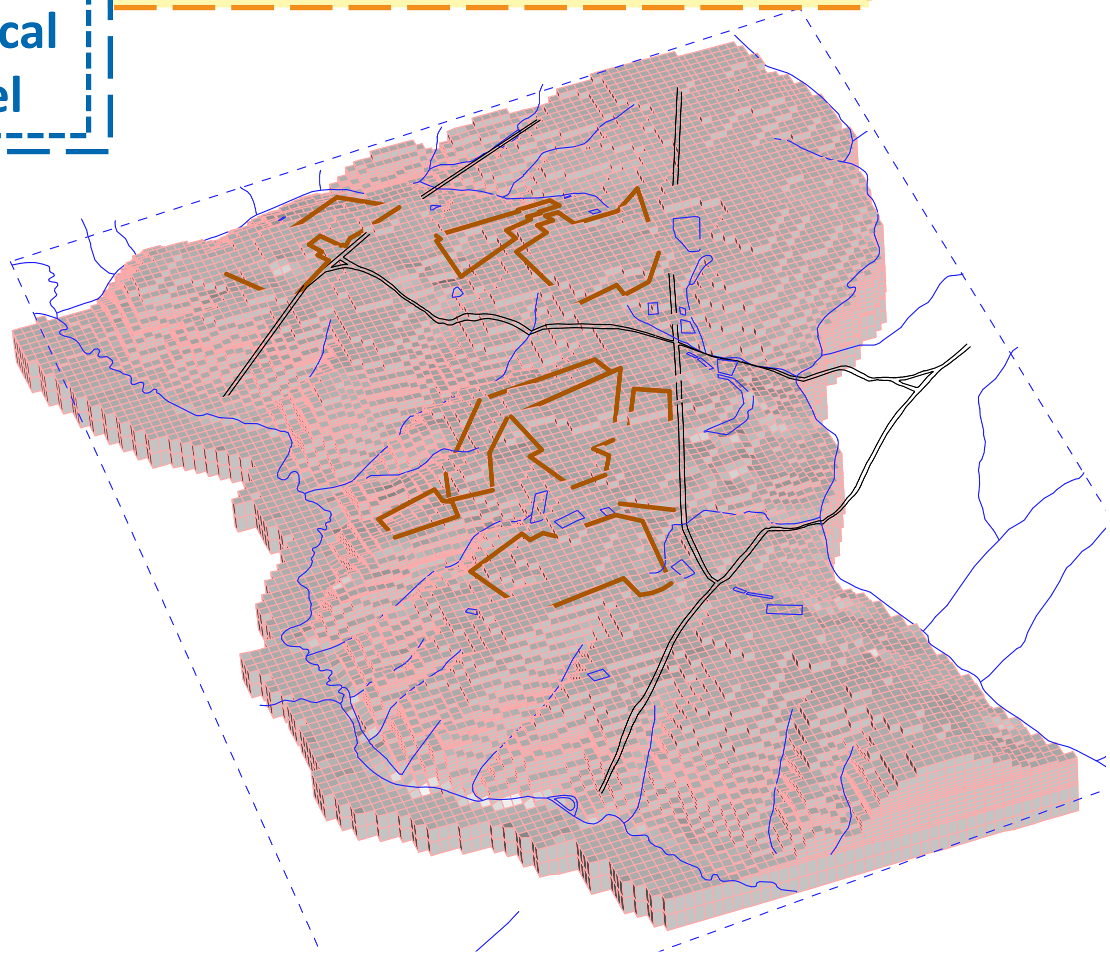
- Flawed Conceptual Model poses greatest Decision risk
- Input (model setup) errors more common than Code errors
- Independent and thorough technical review is critical
- Independent development efforts are highly effective at identifying Model errors

Example: General Separations Area Groundwater Flow Model (Savannah River)



Quality Control check on groundwater flow field Output:

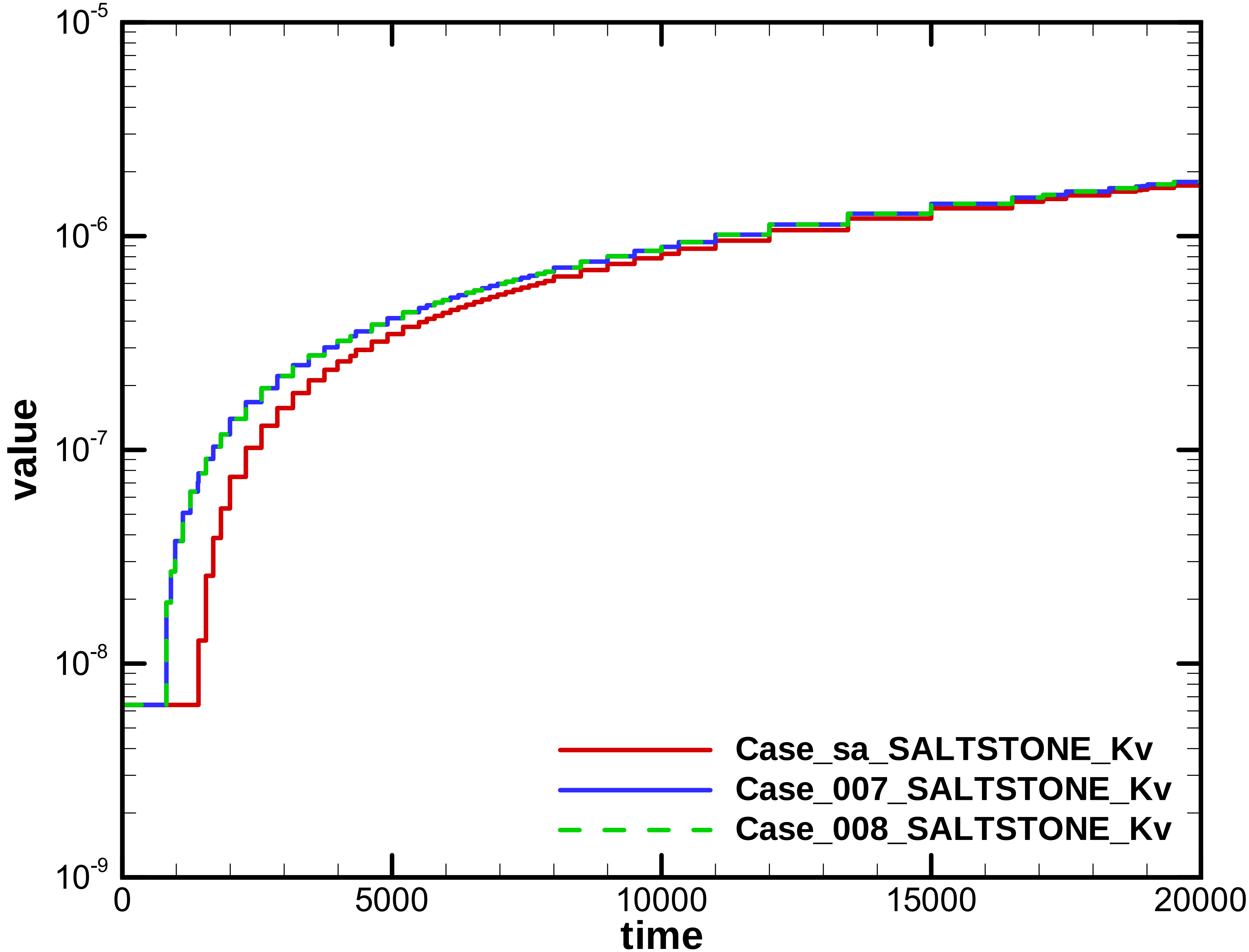
- Confirm steady-state mass balance cell-by-cell and for domain
- Confirm Darcy's Law honored at each cell face
- Confirm simulated and measured water levels and stream baseflows agree



Example: Independent Model Input Summary Tables

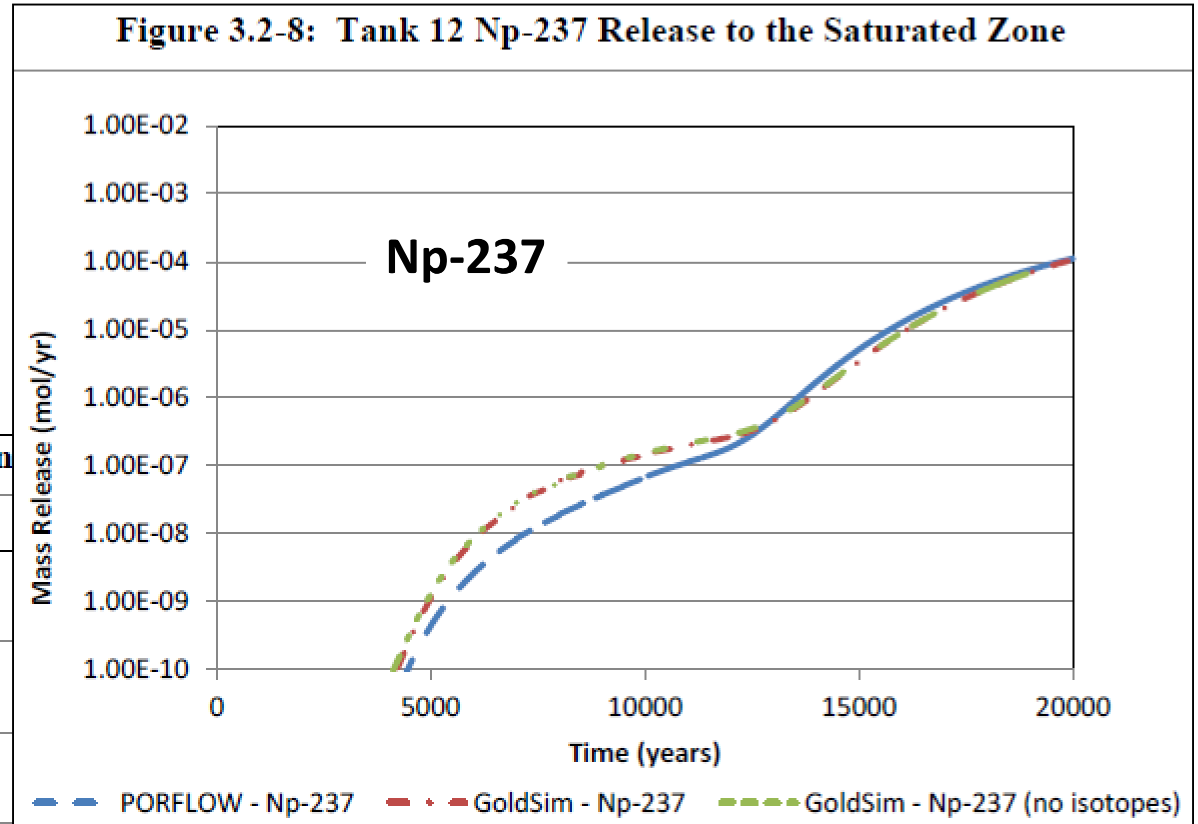
Material	Time Interval	Start	End	Horizontal Conductivity	Vertical Conductivity	Porosity	Density	Water retention
BACKFILL	TI01-TI60	0	100000	7.60E-05	4.10E-05	0.35	2.631	CcBackfill
CLEAN_GROUT	TI01-TI15	0	1106	6.41E-09	6.41E-09	0.58	2.405	fractured_clean
CLEAN_GROUT	TI16	1106	1250	1.68E-06	1.68E-06	0.58	2.405	fractured_clean
⋮								
CLEAN_GROUT	TI20	1690	2113	3.11E-05	3.11E-05	0.58	2.405	fractured_clean
CLEAN_GROUT	TI21-TI60	2113	100000	4.10E-05	4.10E-05	0.58	2.405	fractured_clean
FLOOR	TI01	0	50	1.62E-06	1.62E-06	0.12	2.545	fractured_floor
FLOOR	TI02	50	100	4.86E-06	4.86E-06	0.12	2.545	fractured_floor
⋮								
FLOOR	TI17	1250	1404	8.59E-05	8.59E-05	0.12	2.545	fractured_floor
FLOOR	TI18-TI60	1404	100000	9.11E-05	9.11E-05	0.12	2.545	fractured_floor
ROOF	TI01	0	50	9.07E-07	9.07E-07	0.136	2.558	fractured_roof
ROOF	TI02	50	100	2.71E-06	2.71E-06	0.136	2.558	fractured_roof
⋮								
ROOF	TI15	950	1106	3.71E-05	3.71E-05	0.136	2.558	fractured_roof
ROOF	TI16-TI60	1106	100000	4.10E-05	4.10E-05	0.136	2.558	fractured_roof
WALL	TI01-TI60	0	100000	7.60E-05	4.10E-05	0.35	2.631	CcBackfill

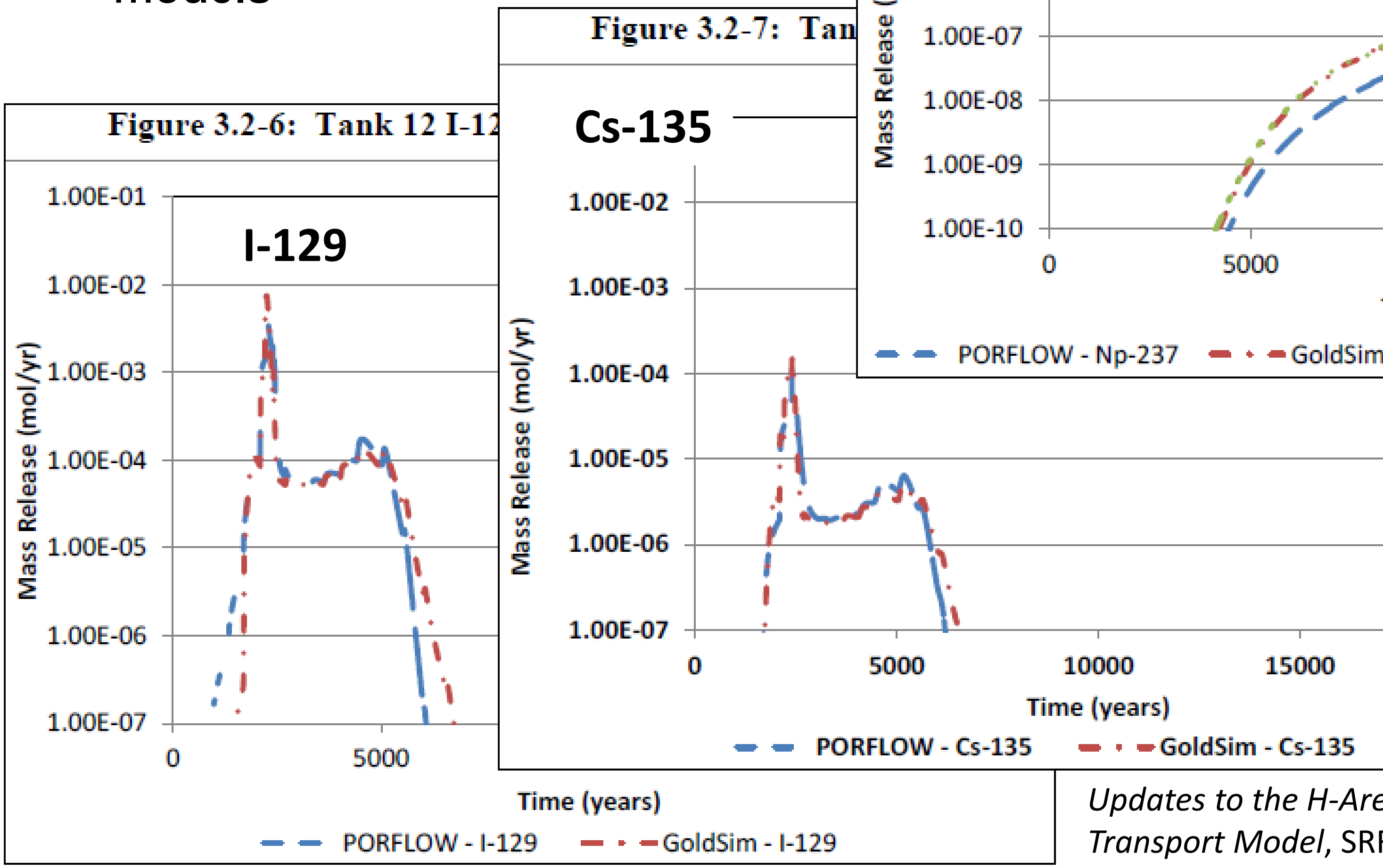
Example: Independent Model Input Summary Graphics



Example: GoldSim and Porflow Benchmarking

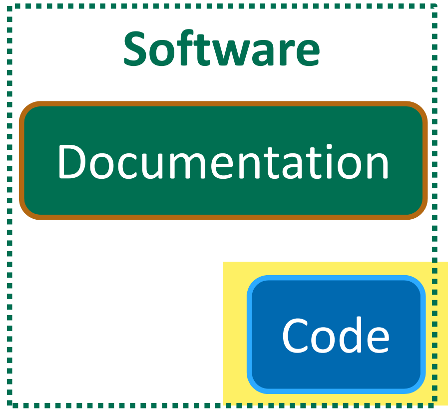
- GoldSim model is an abstracted (simplified) version of Porflow model for UQ/SA
- Independently developed models





Updates to the H-Area Tank Farm Stochastic Fate and Transport Model, SRR-CWDA-2014-00060, Rev. 1, Aug 2015

Quality Assurance (QA) Risks and Remedies



Risks

- Assuming software QA testing addressed the full range of capabilities and conditions
- Assuming software and underlying conceptual models are valid analogues of physical reality
- Assuming good Input will always, or ever, produce good Output

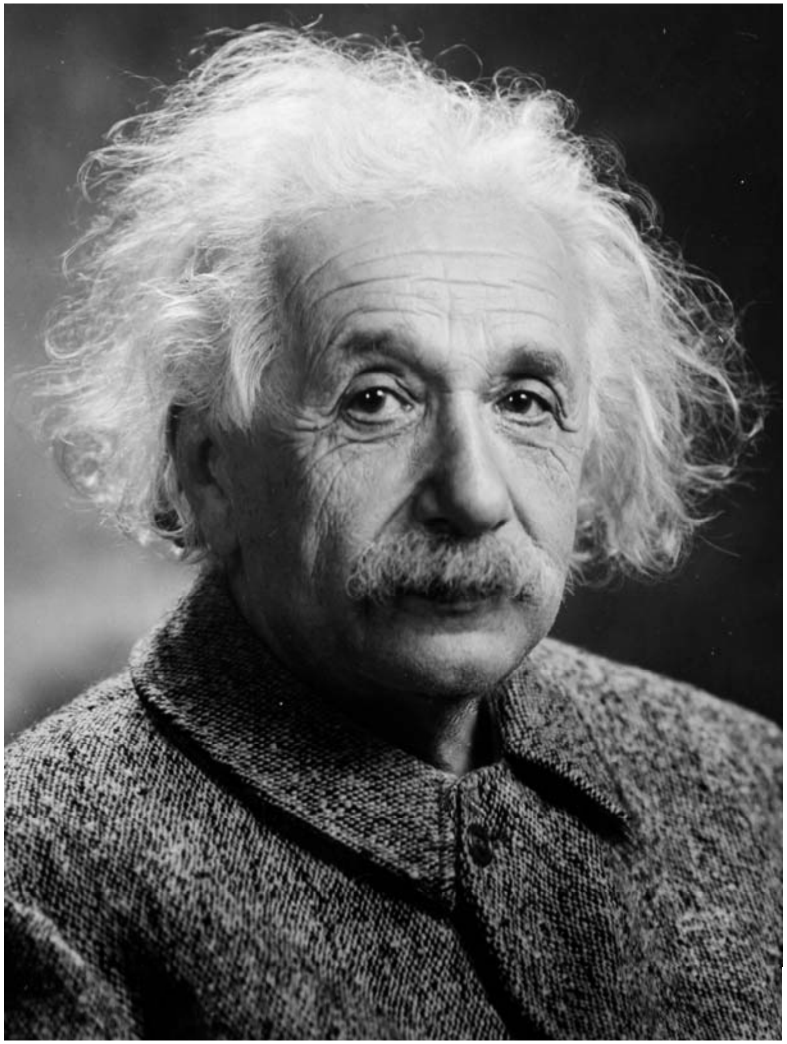
Remedies

- Apply some level of “QC” to each Output
  - e.g. cursory look at every Output
- Code-to-code benchmarking
- Prototypic experiments, field observation, natural analogues

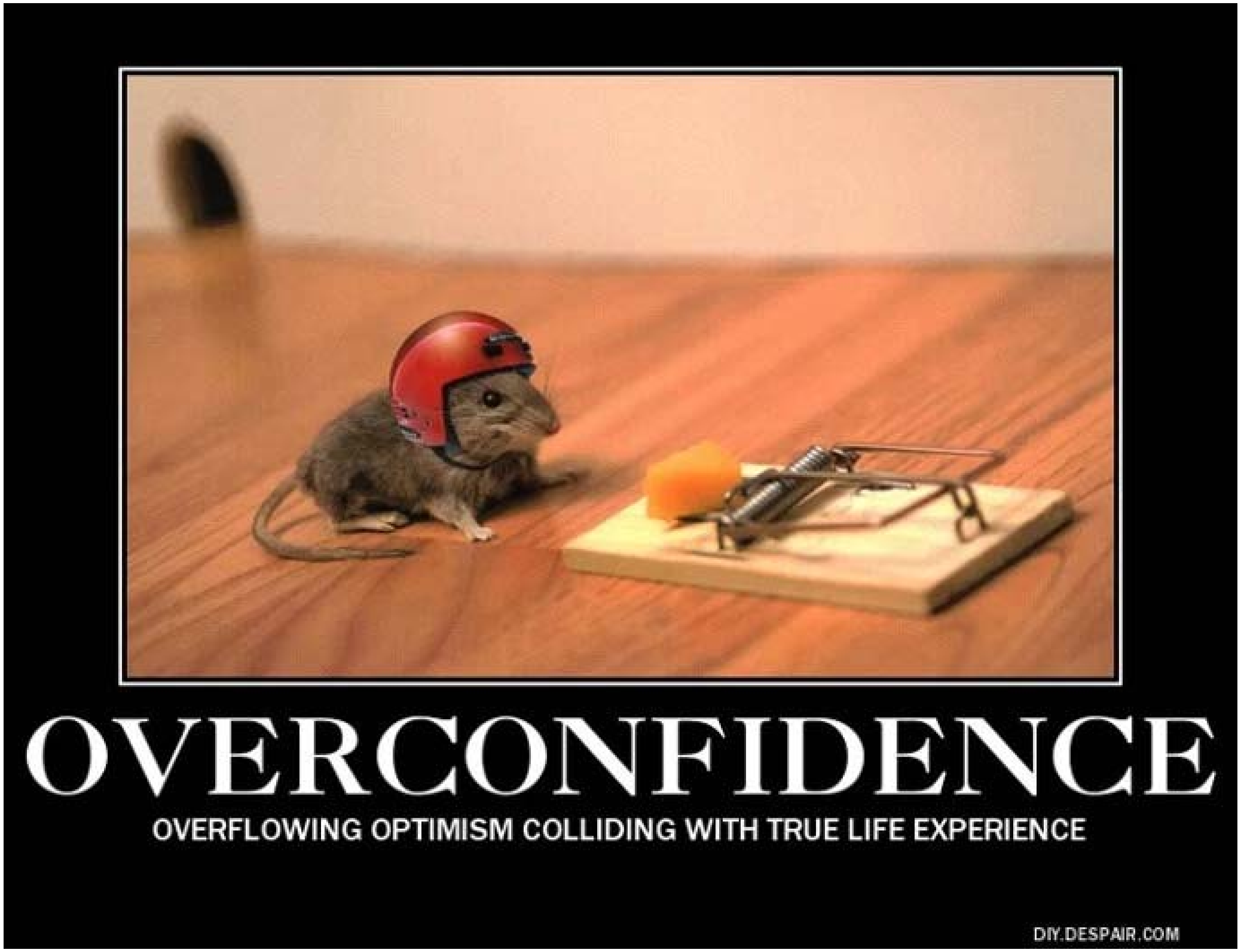
Albert Einstein

"An experiment is something everybody believes, except the person who made it."

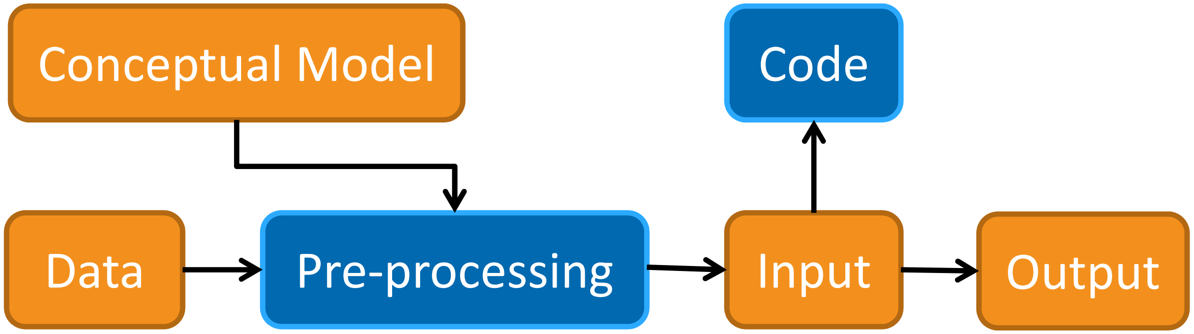
"A theory is something nobody believes, except the person who made it."



"A model is something nobody believes, except the modeler who developed it."



Quality Control (QC) Risks and Remedies



Risks

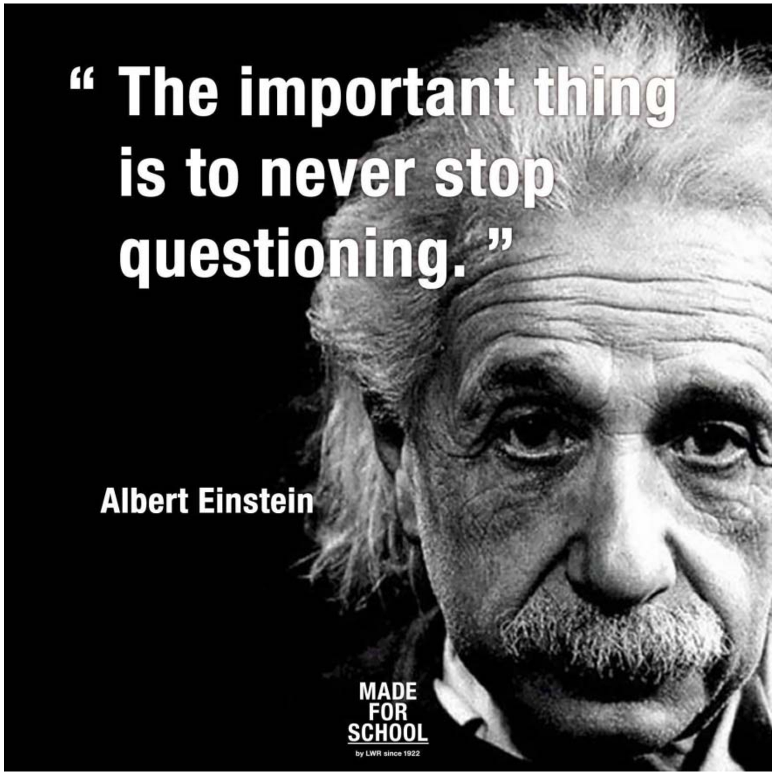
- Assuming custom software … that produced good Output before … will for the next application
- Assuming software and underlying conceptual models are valid analogues of physical reality

Remedies

- Standardize certain pre- and post-processing tools and bring them under formal software QA control
- Avoid complacency in Each Time “QC” checking of model Output
- Prototypic experiments, field observation, natural analogues

Summary Thoughts

- Models are more than Codes
- Model QA is more than Code QA
- Model Output QC (Check Every Time) is a necessary complement to Software QA (Check Once / Use Many)
- Modelers tend to be overconfident in software tools and conceptual models
- Valuable components of Model and Analysis QA
  - Data!
  - Independent modeling efforts
  - Alternative conceptual models
  - Code-to-code benchmarking
  - Skeptical and dedicated expert reviewers



Your thoughts and experiences ?