Quality Assurance and Error Identification for the Community Earth System Model

Allison Baker
I/O & Workflow Applications
Application Scalability and Performance Group

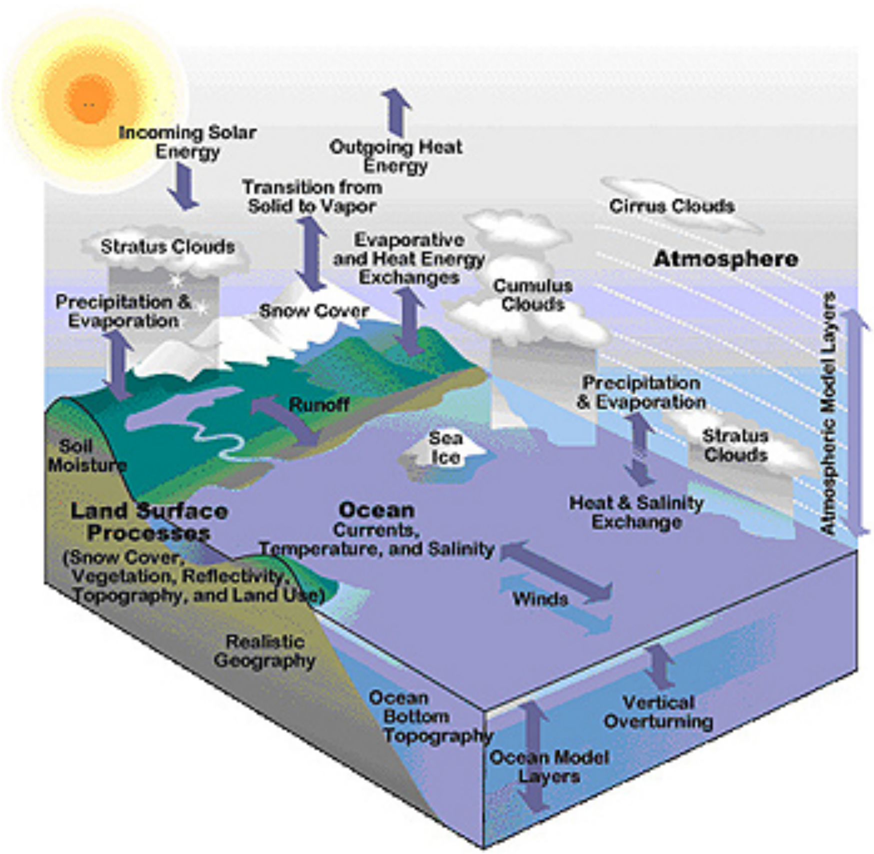
Dan Milroy, Dorit Hammerling, Haiying Xu

Software Correctness for HPC Applications
SC 2017 Workshop
Nov 12,2017

The National Center for Atmospheric Research

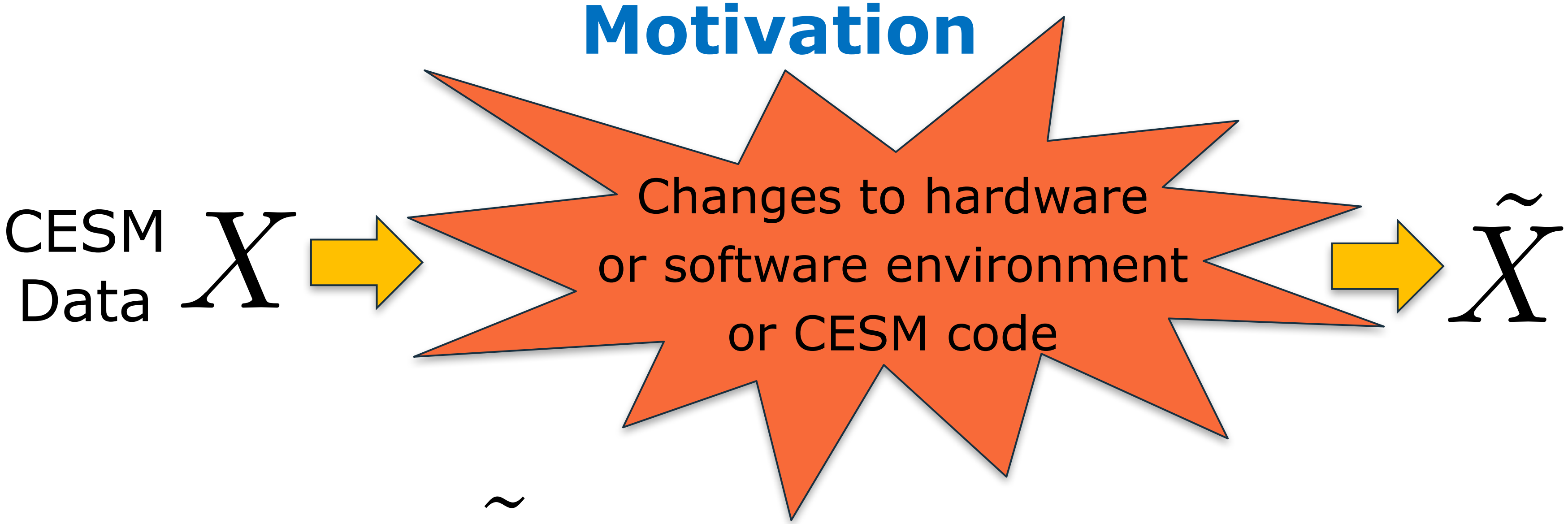
- Boulder, Colorado, USA
- Funded by National Science Foundation (NSF)
- Mission: to understand the behavior of the atmosphere and related Earth and geospace systems

NCAR's Community Earth System Model (CESM)



- past, present and future climate states
- interdisciplinary collaborative effort (led by NCAR)
- ~2M lines of Fortran code (20+ years)
- state of continual development

Motivation



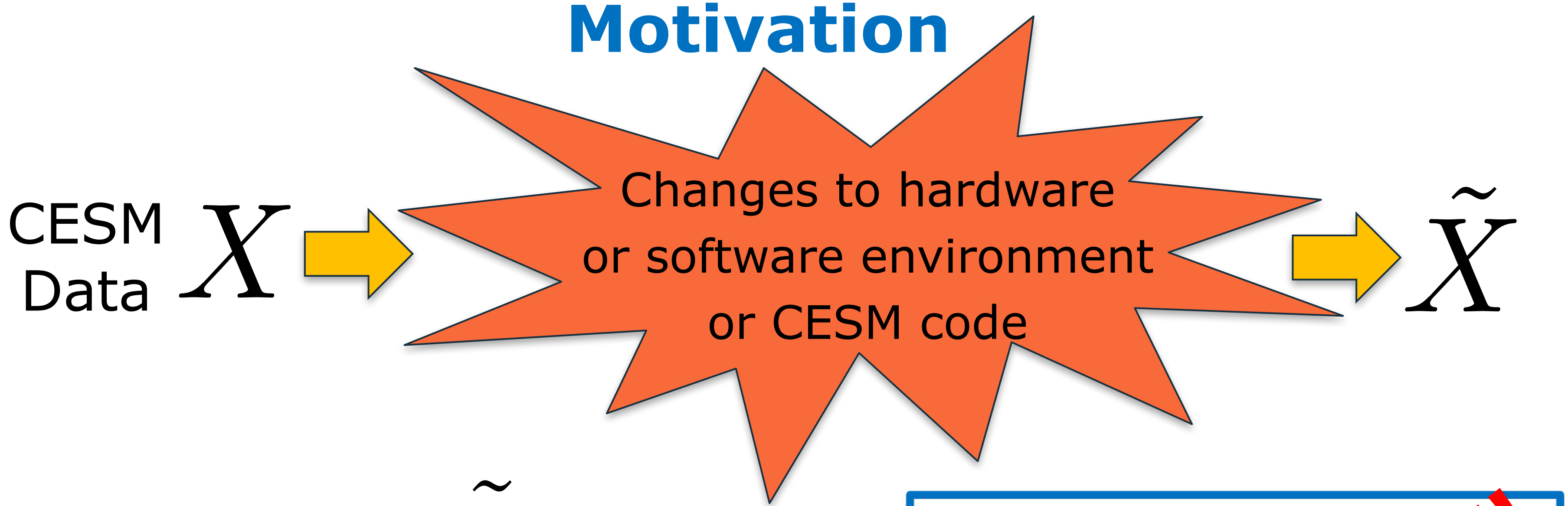
CESM Data X

Changes to hardware or software environment or CESM code

$\tilde{X}$

What if $X \neq \tilde{X}$?

Motivation



What if $X \neq \tilde{X}$?

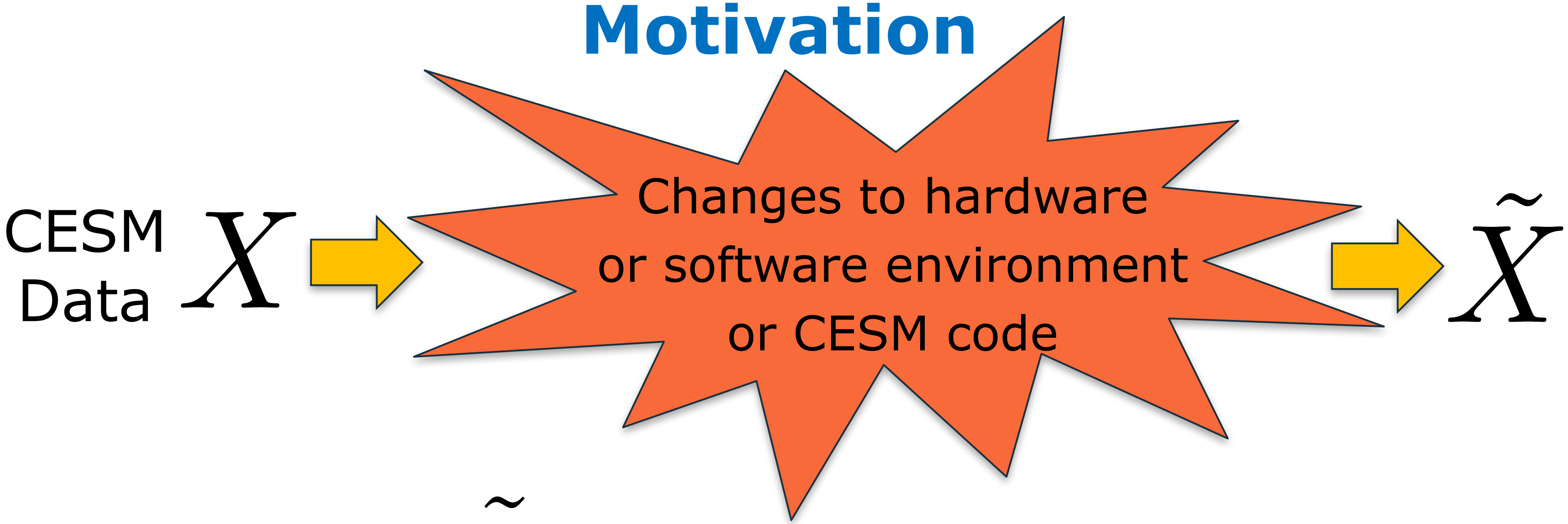
(A) **panic**: must have **bit-for-bit** !!!

CESM results are bit-for-bit reproducible if:

same software,
same compiler and flags,
same MPI,
same parameters,
same initial conditions,
same hardware*,…

TOO RESTRICTIVE!

Motivation



CESM Data X

Changes to hardware or software environment or CESM code

$\tilde{X}$

What if $X \neq \tilde{X}$?

(A) **panic:** must have **bit-for-bit** !!!

(B) **compare LONG simulations:** climate scientist

(C) **automated tool:** ???

Tool → Software Quality Assurance

Insure that changes during the CESM development life cycle do not adversely affect the results!

Question: Is the new result correct?

Wish list: inexpensive, objective, easy-to-use, fast

Challenge: *definition of "correct" or "not climate-changing" ?*

Tool → Software Quality Assurance

Insure that changes during the CESM development life cycle do not adversely affect the results!

Is the new data *statistically distinguishable* from the original?

Wish list: inexpensive, objective, easy-to-use, fast

Challenge: *definition of "correct" or "not climate-changing" ?*

Approach

Q: *Is* X *statistically distinguishable from* $\tilde{X}$ *?*

… allowable error?

Approach: evaluate in the context of climate model’s internal ***variability***

An ensemble of CESM runs:

- “accepted” machine and “accepted” software stack
- *O* (10^{-14}) perturbations in initial temperature
- many variables (use principal components)

Approach

Q: *Is X statistically distinguishable from $\tilde{X}$?*

… allowable error?

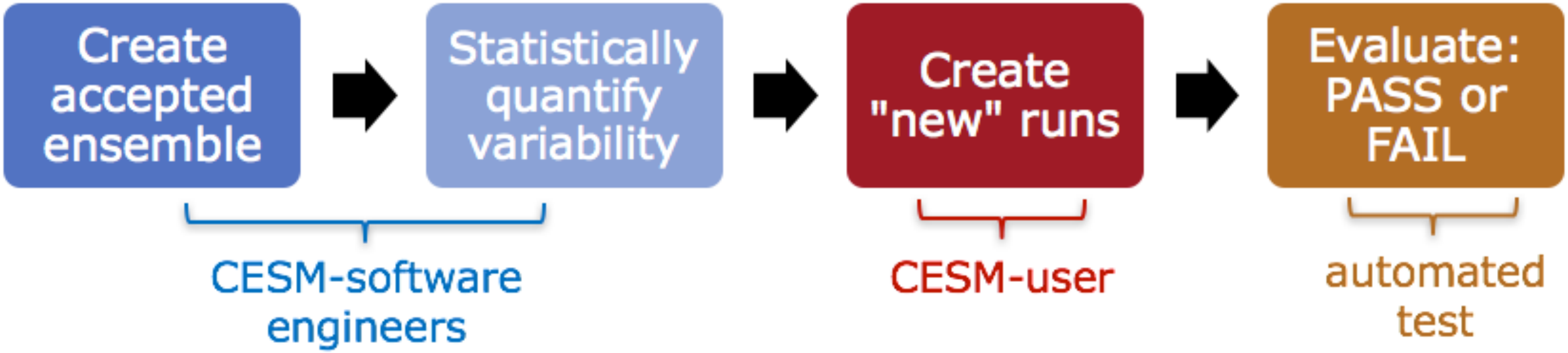
Approach: evaluate in the context of climate model’s internal ***variability***

An ensemble of CESM runs:

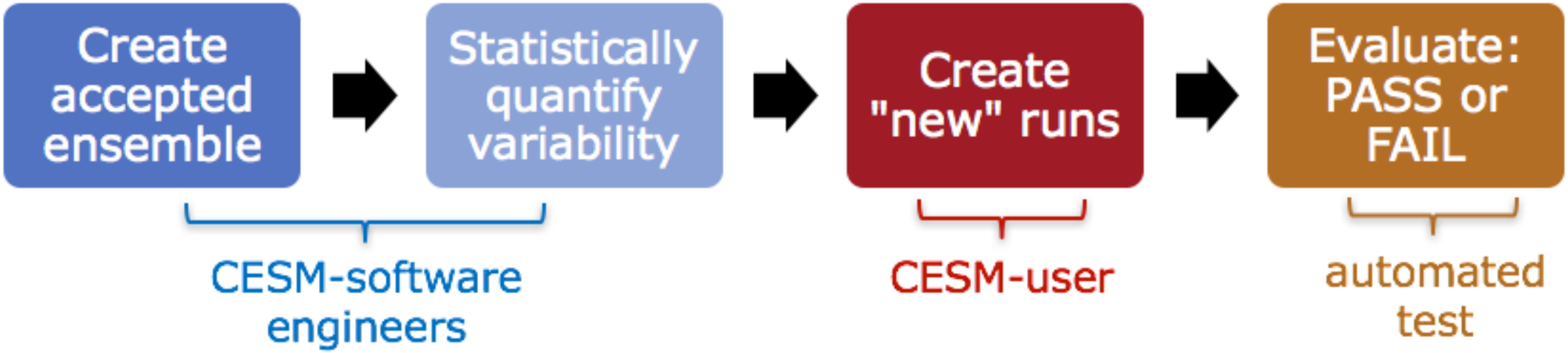
- “accepted” machine and “accepted” software stack
- *O* (10^{-14}) perturbations in initial temperature
- many variables (use principal components)

*yields an “**accepted**” statistical distribution that can be used to evaluate “new” runs*

Ensemble Consistency Test (ECT)



Ensemble Consistency Test (ECT)



Create accepted ensemble → Statistically quantify variability → Create "new" runs → Evaluate: PASS or FAIL

CESM-software engineers

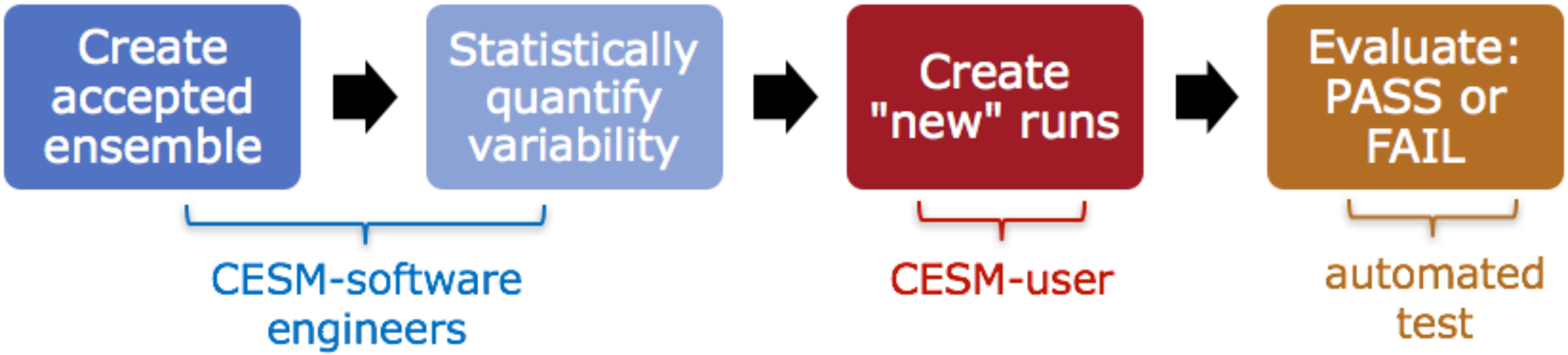
CESM-user

automated test

Highlights:

- automated Python tool
- objective, user-friendly
- rapid feedback for model developers
- suite of tools: atmosphere, land, ocean, sea ice

Ensemble Consistency Test (ECT)



Create accepted ensemble → Statistically quantify variability → Create "new" runs → Evaluate: PASS or FAIL

CESM-software engineers

CESM-user

automated test

Highlights:

- automated Python tool
- objective, user-friendly
- rapid feedback for model developers
- suite of tools: atmosphere, land, ocean, sea ice

Tool	Target Modules
CAM-ECT	CAM, CLM
POP-ECT	POP, CICE
UF-CAM-ECT	CAM, CLM

How well does CESM-ECT work?

- modifications *expected* to be climate-changing *fail*
  - e.g. relative humidity, dust emissions, CO_2 levels
- modifications *not expected* to be climate changing *pass*
  - e.g., threads, -O0, compiler version, code rearrangement
- option when bit-for-bit reproducibility is not possible:
  - new algorithms, solvers, compiler options, hardware technologies

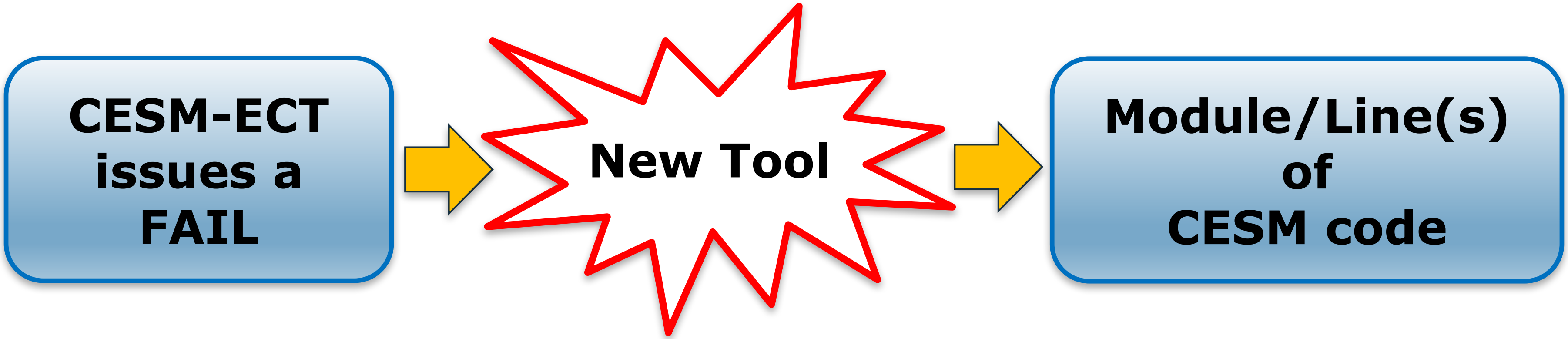
...but this is a coarse-grain test

Fine-grain tool: root cause

Identify/understand the reason for the inconsistency!

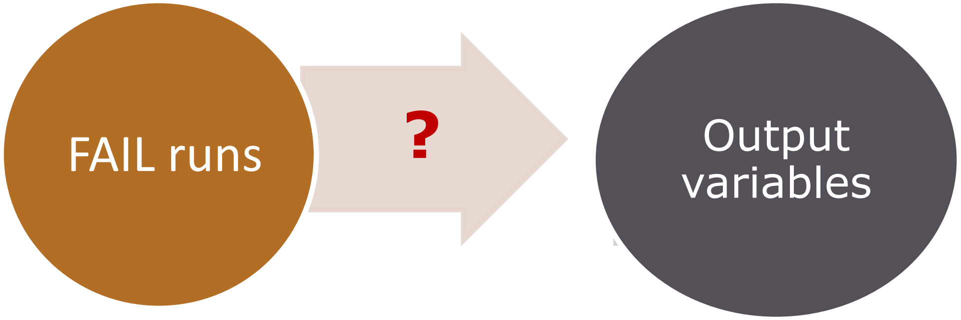
CESM-ECT "fail" :

- *currently*: principal component information…
- *in progress*: give helpful information!
  (variable(s), module(s), etc.)



Fine-grain tool: root cause

Motivation: inconsistency with FMA (Fused Multiply-Add)



Which output variables contributed to the failure?

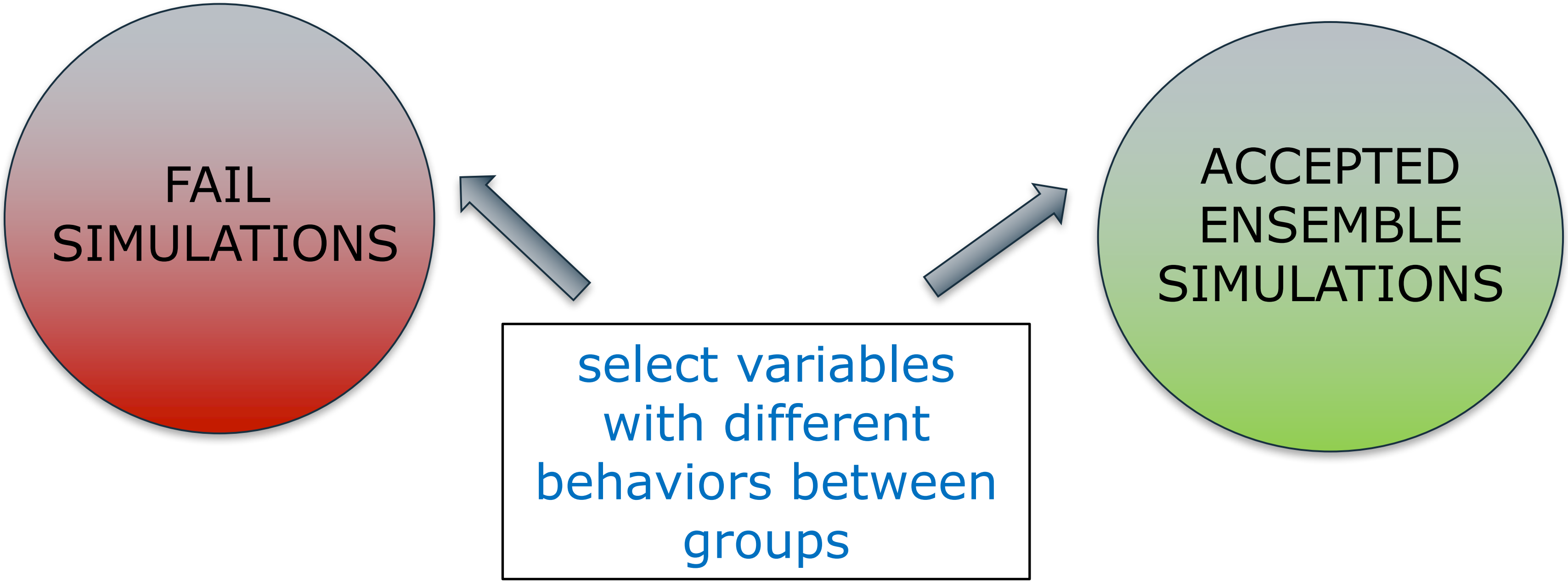
principal components => output variables

Initial (slow): systematic exclusion of variable combinations
(redo PCs/test)

Better (ML): logistic regression + variable selection

Fine-grain tool: root cause

logistic regression + variable selection



FAIL SIMULATIONS

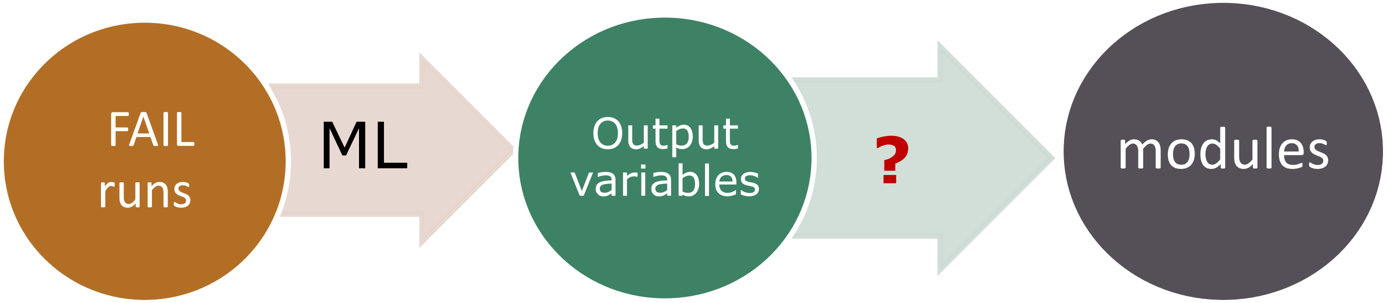
ACCEPTED ENSEMBLE SIMULATIONS

select variables with different behaviors between groups

- simulations are 9 time steps (cheap)
- ~30 FAIL runs, ~350 ensemble runs
- Scikit-learn: randomized logistic regression

Fine-grain tool: root cause

Which CESM modules affect output variables?

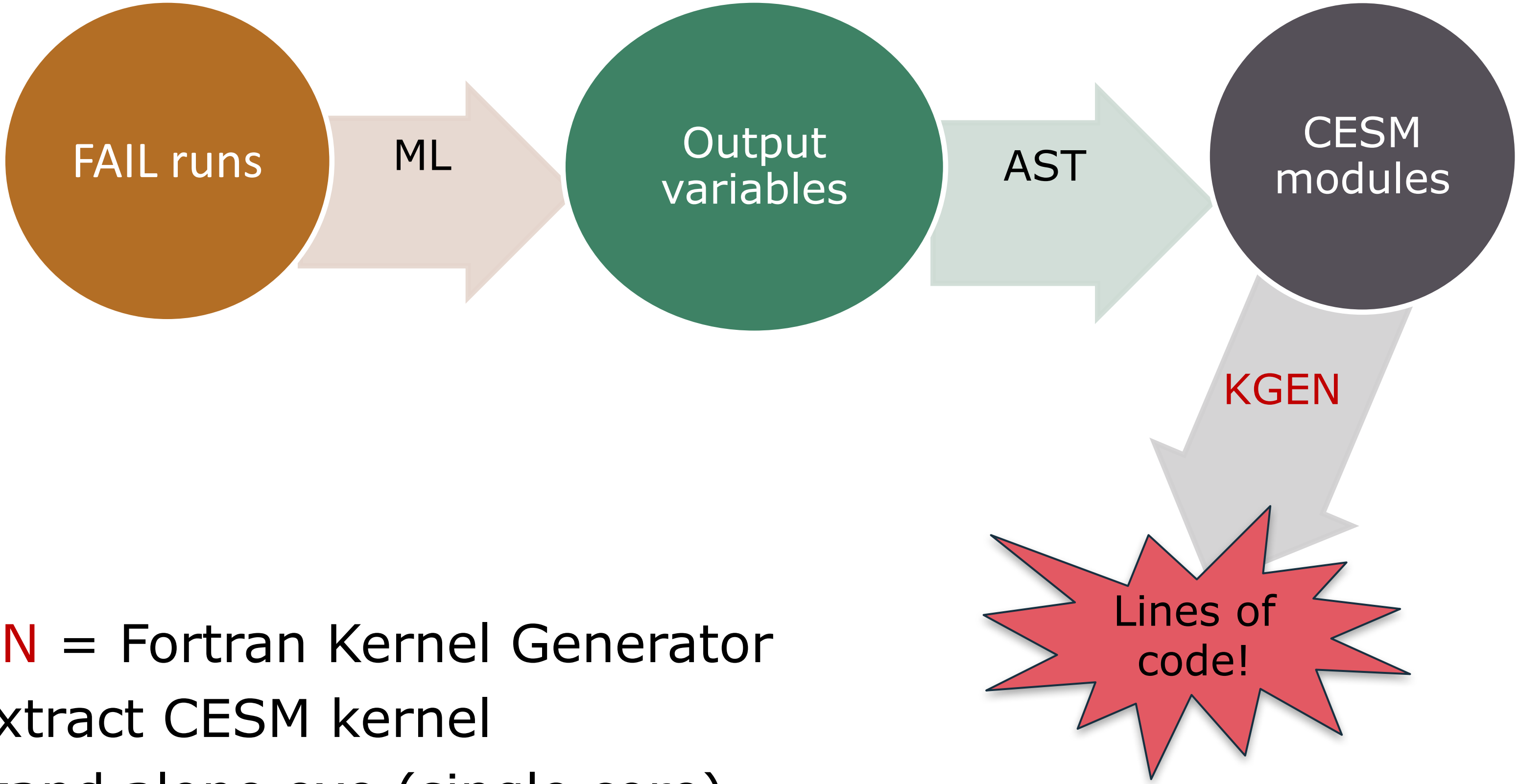


Manual: talk to climate scientists

Automated: abstract syntax tree for CESM

- graph structure of source code
- non-trivial: ~2M lines of complex Fortran code
- *in progress*

Fine-grain tool: root cause



KGEN = Fortran Kernel Generator

- extract CESM kernel
- stand alone exe (single core)
- identify differences in internal variables

Concluding remarks

- *improve quality assurance & error identification in CESM!*
  - large and complex code
  - minor differences => differences in simulation output
- ensemble consistency approach
  - objective, user-friendly
  - port-verification (new CESM-supported architectures)
  - uncovered multiple errors in code and hardware
- cause of statistical inconsistency
  - nearly complete!

Thanks!

abaker@ucar.edu