# Some thoughts on the future of PETSc

Barry Smith

PETSc still does what it does well, but ...

# Early Decisions (1994)

- All-in on MPI
  - Single-core compute nodes (no explicit thread support ??)
- C ??
- Native Fortran binding
- Delay decisions to runtime - "because I don't know what the correct choices are"
  - Dynamic Object Orientation
  - Runtime polymorphism (no templates/code generation ?? )
    - All objects delegate
  - Options Database
- Little support for array operations (DMDA, PetscSection)
- No support for user thread parallelism with PETSc MPI parallelism ??
- Easily extendable to new methods, solver families etc

# Last five years' time wasters

The race to keep up with hardware/software changes slows adding new features

- The need to focus on utilizing GPUs left little time to add new features, and solver types.
- Supporting portability, language/compiler changes (picky C++/for GPUs)

# Moving Forward

- Simulation within outer loops (we've been saying this for many years but making little progress)
  - simulation-constrained optimization (introduces MPI unfriendly long-and-skinny data structures)
  - simulation-based probability and statistics (tools are often written in custom languages)
- Simulation plus ML
  - data structure (tensors to PETSc) interoperability
  - batching in PETSc
- The above requires derivatives (algorithmic, automatic differentiation, ..), except for TSAdjoint, PETSc provides little support
- A more systematic approach to hardware portability
  - Abstractions for data structures and operations on data structures?

Can we engage people from related communities to assist in moving PETSc forward?