VERIFICATION AND VALIDATION OF MOM6

- Alistair Adcroft
- Robert Hallberg
- **Marshall Ward**

2020 June 8

`https://marshallward.org/mom6vv`

Verification

> *Am I building the product right?*

Validation

> *Am I building the right product?*

Barry Boehm, *Software Risk Management* (1989)

VERIFICATION

What are the *design specifications* of my model?

- Does it compile on target platforms?
- Are the equations dimensionally consistent?
- Does parallelization change the answers?

Verification is the confirmation of design specifications.

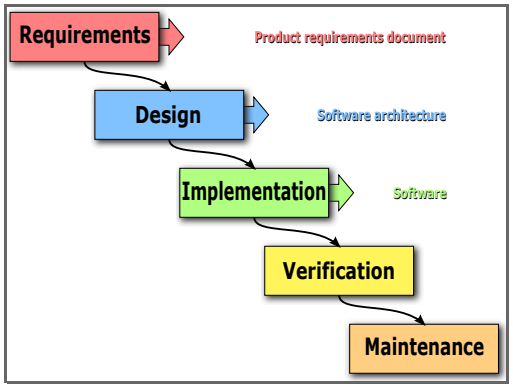
VALIDATION

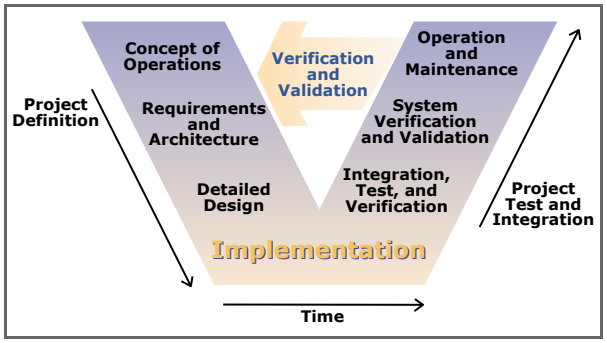
Does our model meet operational needs?

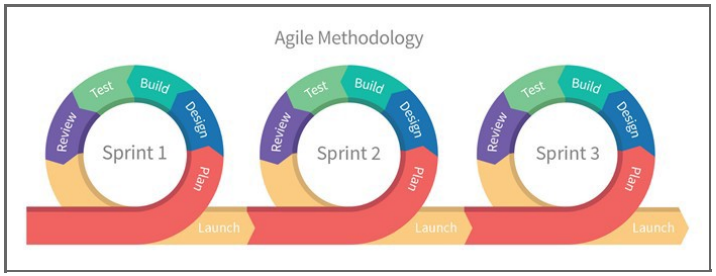
- Does it produce realistic simulations?
- Are relevant physical features present?
- Can I reproduce my old simulations?

Validation is an assessment of the final product.

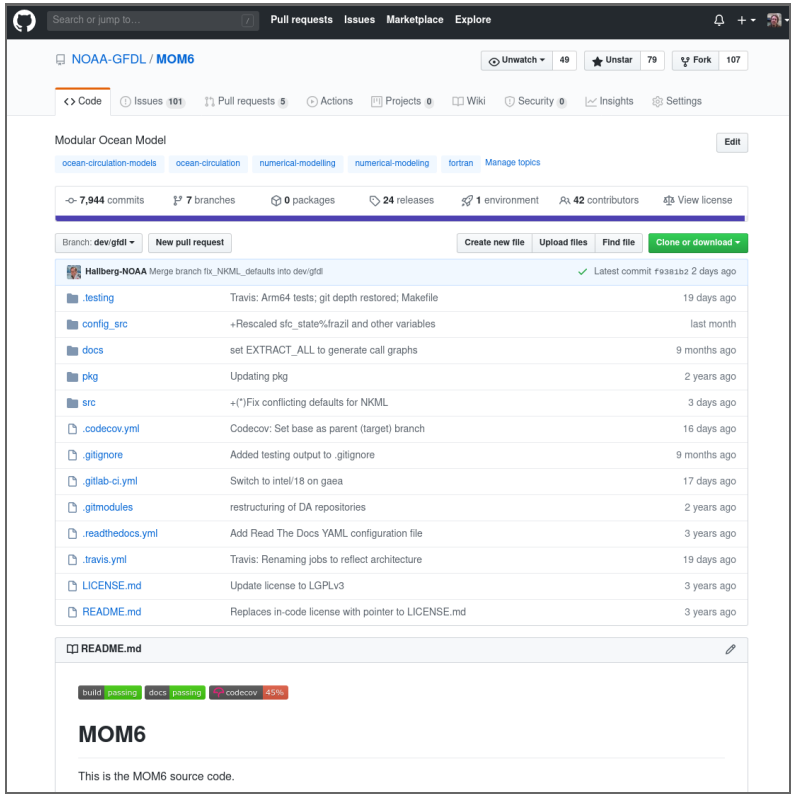
V&V IN DEVELOPMENT



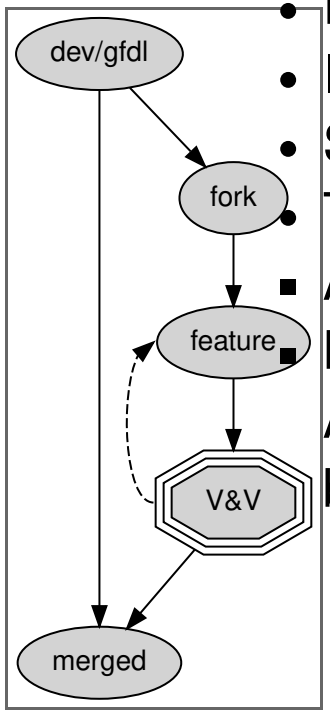




MOM6 DEVELOPMENT

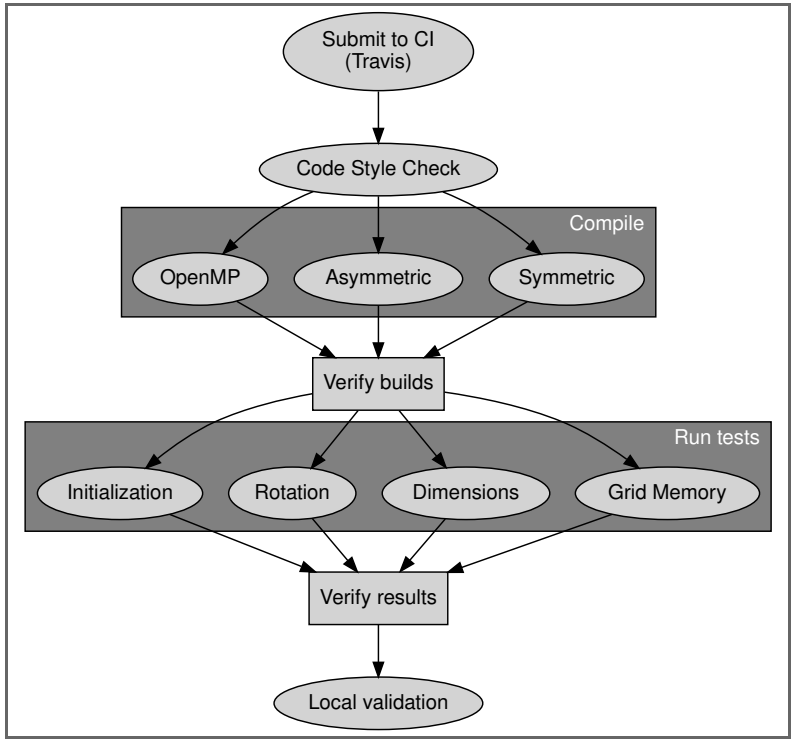


V&V IN MOM6



- Fork from a community repository
- Implement feature
- Submit as Pull Request™ (PR)
- Trigger V&V events
  - Automated verification
  - Manual validation
- All contributions must pass verification and validation before merge.

MOM6 VERIFICATION



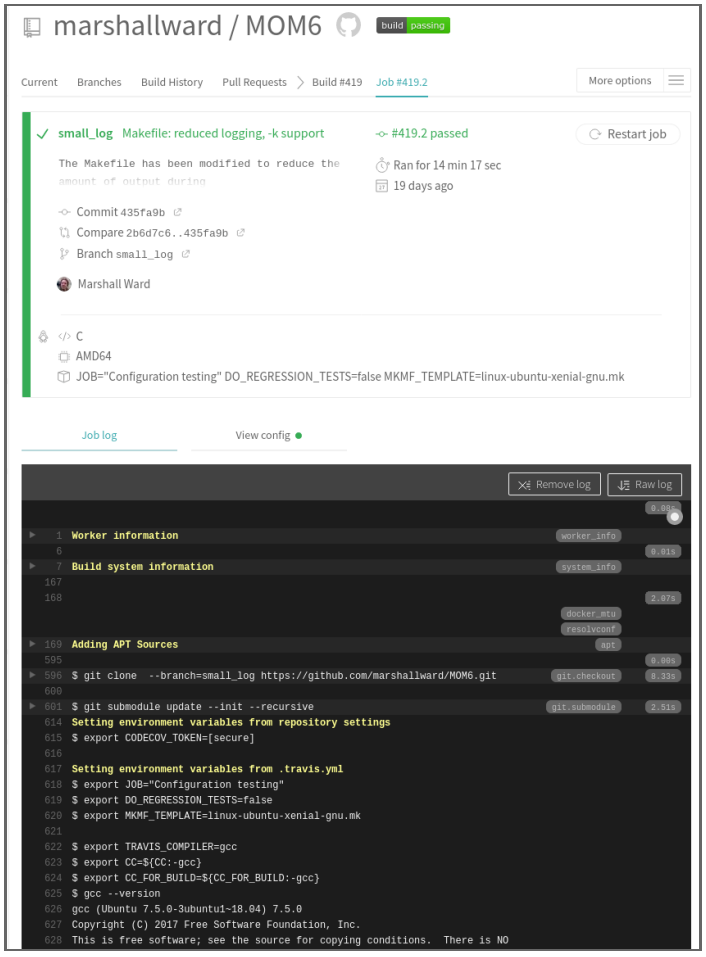
All changes sent to CI (Travis) for verification

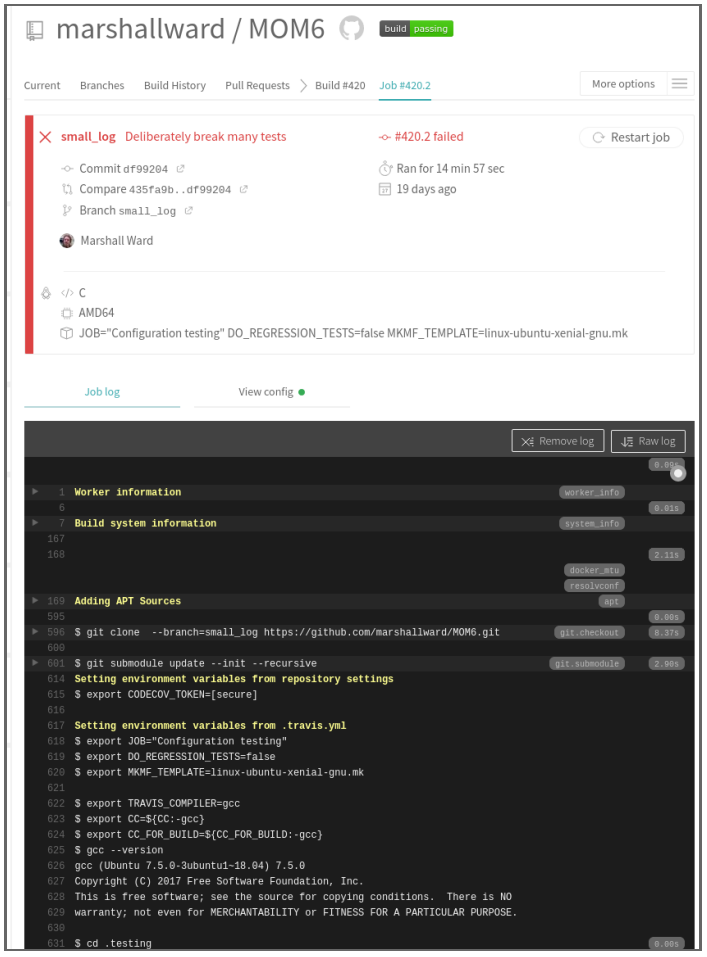
VERIFICATION TESTS

Test	Description
grid	Symmetric/Asymmetric memory grids
layout	1×1 and 2×1 domain decomposition
rotation	Index rotation
restart	Restart at mid-run
repro	Optimized reproducible mode
openmp	OpenMP (single-thread)
nan	NaN array initialization
dim	Dimensional scaling

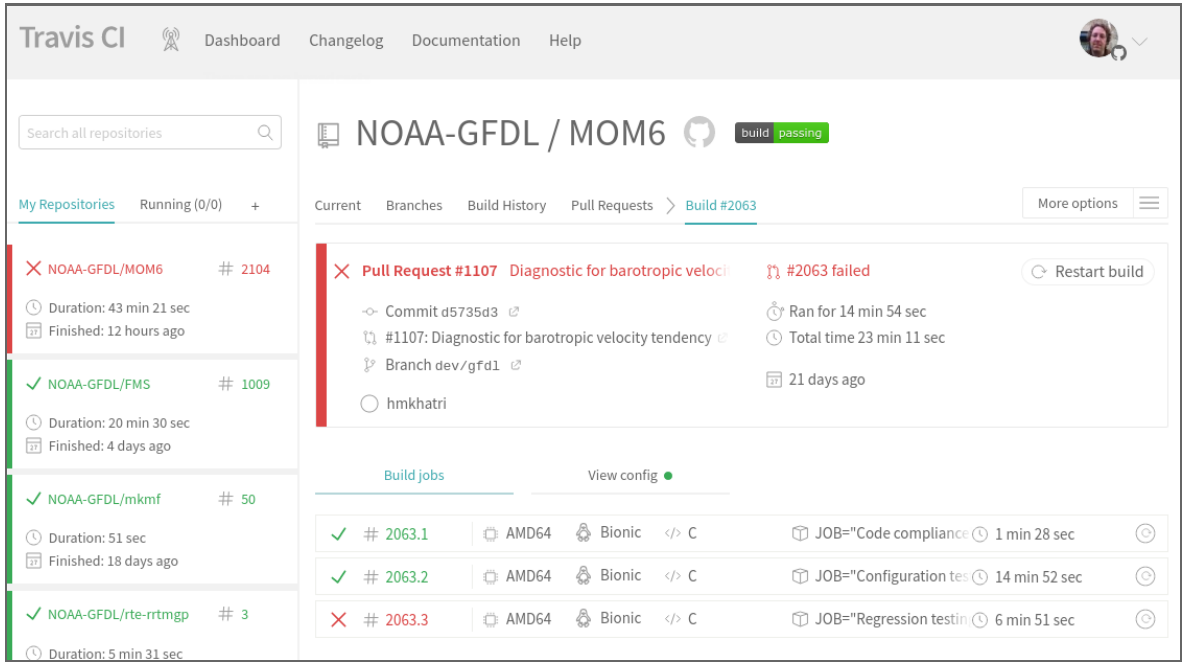
Each test requires bit reproducibility

TESTS IN ACTION



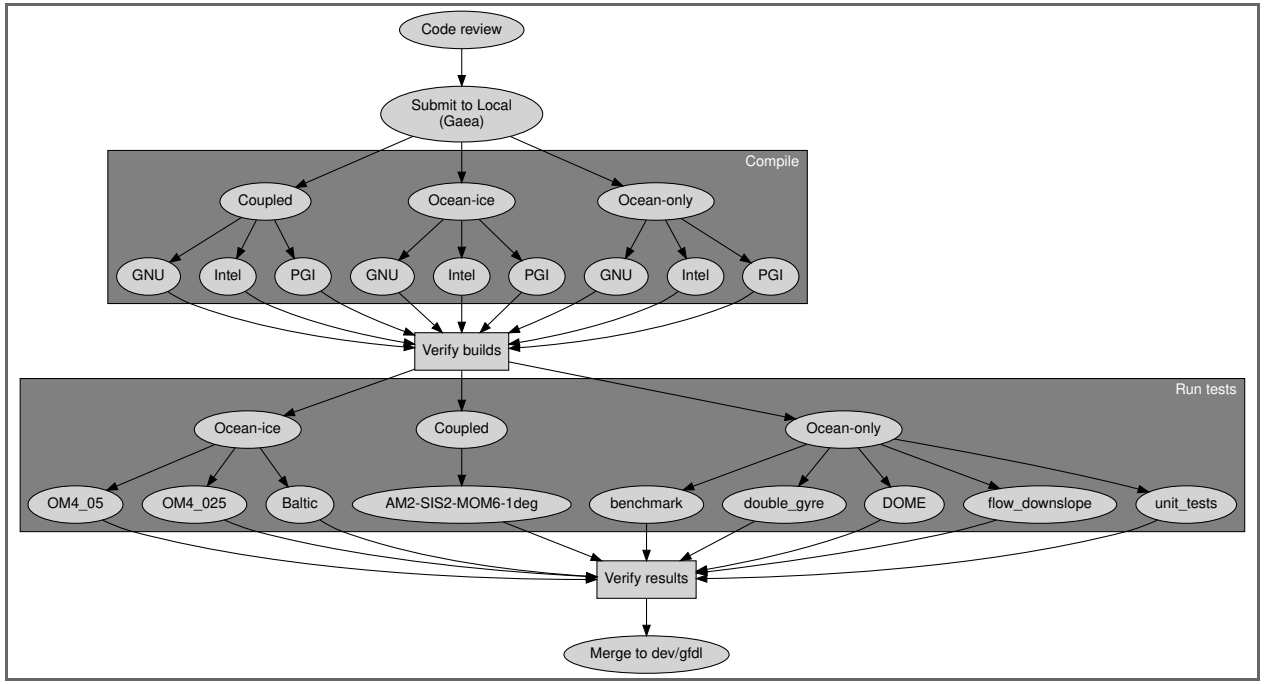


REGRESSIONS



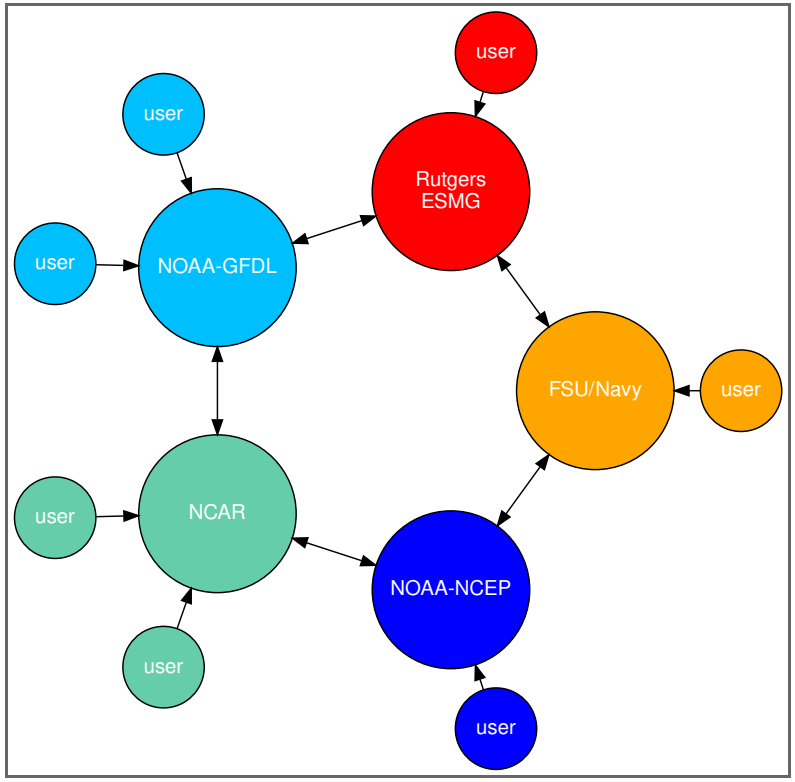
What if we want answers to change?

MOM6 VALIDATION



Current validation includes over 60 tests

HUB VALIDATION



SOLUTION VERIFICATION

`ocean.stats`:

```
Step,        Day,  Truncs,       Energy/Mass,
Maximum CFL,  Mean Sea Level,  Total Mass,  Mean Salin,
Mean Temp, Frac Mass Err,   Salin Err,    Temp Err
           [days]                 [m2 s-2]
[Nondim]        [m]              [kg]          [PSU]
[degC]        [Nondim]         [PSU]         [degC]
   0,        0.000,     0, En 7.2161166068132286E-27,
CFL  0.00000, SL  1.8190E-12, M 1.39637E+20, S 35.0000,
T 13.3483, Me  0.00E+00, Se  0.00E+00, Te  0.00E+00
  12,        0.500,     0, En 2.7781004671136538E-04,
CFL  0.00011, SL  1.1369E-12, M 1.39637E+20, S 35.0000,
T 13.3484, Me -6.09E-17, Se -3.90E-15, Te -1.17E-15
  24,        1.000,     0, En 2.7734897826598717E-04,
CFL  0.00014, SL  1.8190E-12, M 1.39637E+20, S 35.0000,
T 13.3486, Me  2.89E-17, Se  8.80E-17, Te -2.88E-16
```

Based on global metrics (energy, mass, etc)

DIAGNOSTIC VERIFICATION

`chksum_diag`:

```
u-point: mean=   1.1239682303793666E-04 min=
-6.7187595818683776E-03 max=   3.3480219779204019E-02
ocean_model-u
u-point: c=     21851 ocean_model-u
v-point: mean=   1.2076392816784489E-03 min=
-8.3469699425156359E-03 max=   6.8420831486068704E-03
ocean_model-v
v-point: c=     18606 ocean_model-v
h-point: mean=   3.6490088139048595E+02 min=
9.9999999999999915E-04 max=   5.6265092225099863E+02
ocean_model-h
h-point: c=     18673 ocean_model-h
```

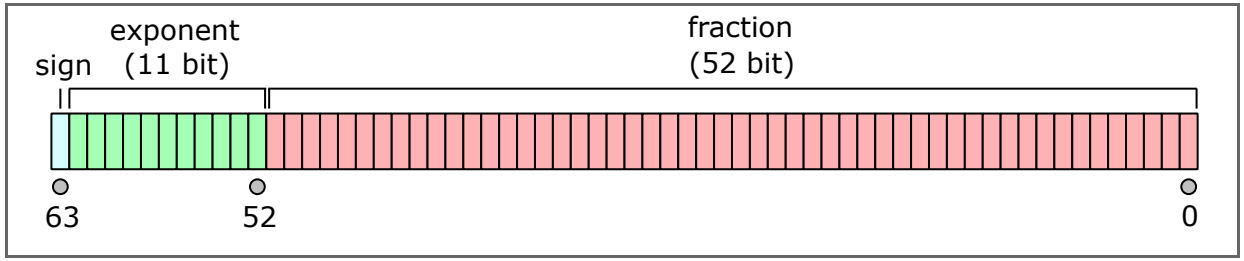
Min, max, mean, bitcount for every diagnostic

BIT REPRODUCIBILITY

Verification requires bit reproducibility

Identical code and input, different math libraries (c/o **Foone**)

FLOATING POINT REVIEW



$$\phi \equiv (-1)^{s} \times 2^{M} \times (1 + \alpha)$$

- Smallest fractional diff: $2^{-52} \approx 2.2 \times 10^{-16}$
- 17 digits to uniquely specify a result

ADDITION ASSOCIATIVITY

What is $10^{-16} + 1 - 1$?

$$(10^{-16} + 1) - 1 = 0$$
$$10^{-16} + (1 - 1) \equiv 10^{-16}$$

Residuals below 2×10^{-16} may be lost.

MORE ADDITION EXAMPLES

Let $s = 1 + 2 \times 10^{-16}$. What is $(s + 1) - 1$?

$$s + 1 = 2$$
$$(s + 1) - 1 = 1 \neq s$$

Manipulation of s shifted the least resolvable value.

MULTIPLICATION ASSOCIATIVITY

If $a = b = 1.5$, and $c = 1 + 2^{-52}$, then

$$(a \times b) \times c \equiv 2.25 + 2^{-51}$$
$$a \times (b \times c) \equiv 2.25 + 2^{-50}$$

(Actual results depend on rounding rules)

SAMPLE PROGRAM

```
program rounding
  use iso_fortran_env, only : real64
  implicit none

  real(kind=real64) :: a, b, c

  a = 1.5
  b = 1.5
  c = 1.0000000000000002_real64

  print '(a, es23.17)', "(a * b) * c = ", (a * b) * c
  print '(a, es23.17)', "a * (b * c) = ", a * (b * c)
end program rounding
```

INTEGRITY OF PARENTHESES

V&V requires integrity of parentheses
GCC Fortran

```
gfortran -fprotect-parens ...    # default
gfortran -Ofast ...              # Sets -fno-protect-
parens
```

Intel Fortran

```
ifort -assume protect-parens     # Not default
```

Note: Fortran requires this!

DIVISION PERFORMANCE

Minimize division operations:

```
x = a / b / c             ! Bad
x = a / (b * c)           ! Good

y = 1. / (1. + 1./c)      ! Bad
y = c / (c + 1.)          ! Good
```

Store common divisions:

```
I_dx = 1.0 / dx
dudx = I_dx * (u(i+1) - u(i))
dvdx = I_dx * (v(i+1) - v(i))
```

Divisions are slower and more unpredictable

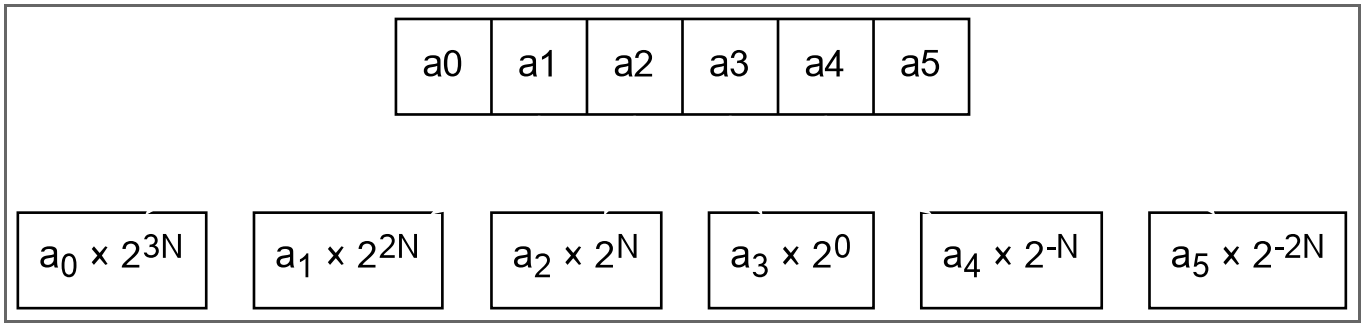
PARALLEL SUMMATION

How to compute reproducible means or global sums?

- Enforce ordering

$$\sum \phi = (\phi_1 + (\phi_2 + (\phi_3 + \dots)))$$

- Fixed-precision arithmetic



ASSOCIATIVE SCALING

Recall the floating point format

$$\phi \equiv (-1)^s \times 2^M \times (1 + \alpha)$$

Power-of-two multiplication is associative

$$2^N \times \phi \times 2^{-N} \equiv \phi$$

DIMENSION SCALING

Fields rescaled by dimensions should be invariant

$$u^{n+1} = u^n + \Delta t \times \mathcal{F}$$
$$2^{L-T} u^{n+1} = 2^{L-T} u^n + 2^T \Delta t \times 2^{L-2T} \mathcal{F}$$

DIMENSIONAL FACTORS

Unit	Scaling	Name
s	T	Time
m	L	Horizontal length
m	H	Layer thickness
m	Z	Vertical length
kg/m3	R	Density
J/kg	Q	Enthalpy

DEFINING DIMENSIONS

Input parameters

```
call get_param(... , "DT", ... , scale=US%s_to_T)
```

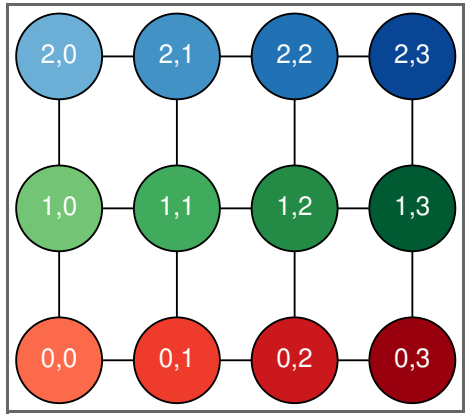
Explicit constants

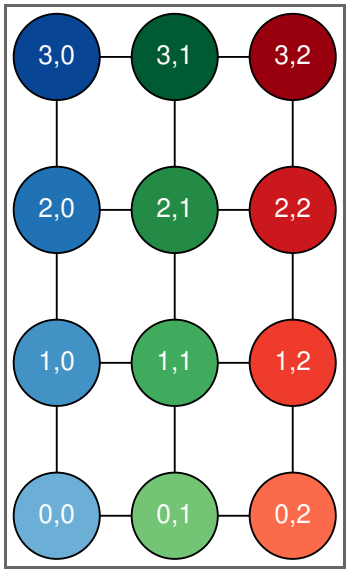
```
eps_vel = 1.0e-10 * US%m_s_to_L_T
ustar = 0.01 * US%m_to_Z * US%T_to_s
```

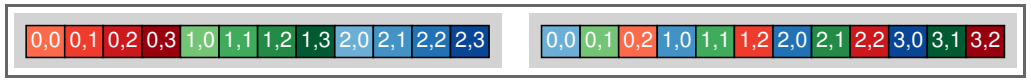
Diagnostic registration

```
call register_diag_field(..., "u", ... ,
conversion=US%L_T_to_m_s)
```

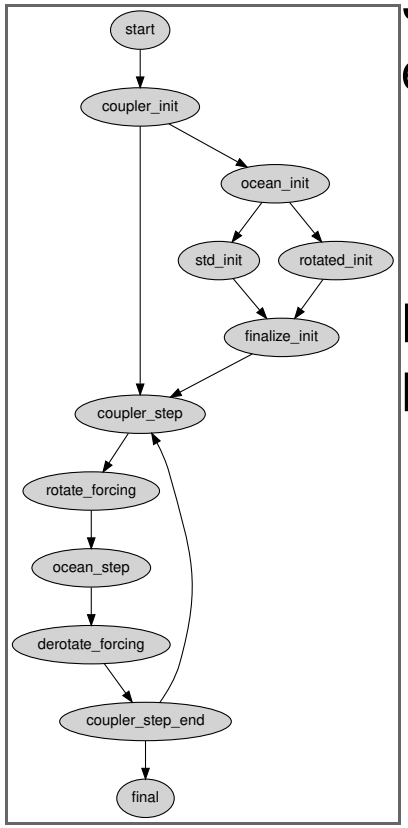
INDEX ROTATION







ROTATION INVARIANCE



Solutions must be invariant to **index rotation**, e.g.:

$$\phi(i', j') = \phi(j, N - i)$$

Both *fields* and *coordinates* are remapped.

Note: u and v are velocities along i and j!

ROTATIONAL CONSISTENCY

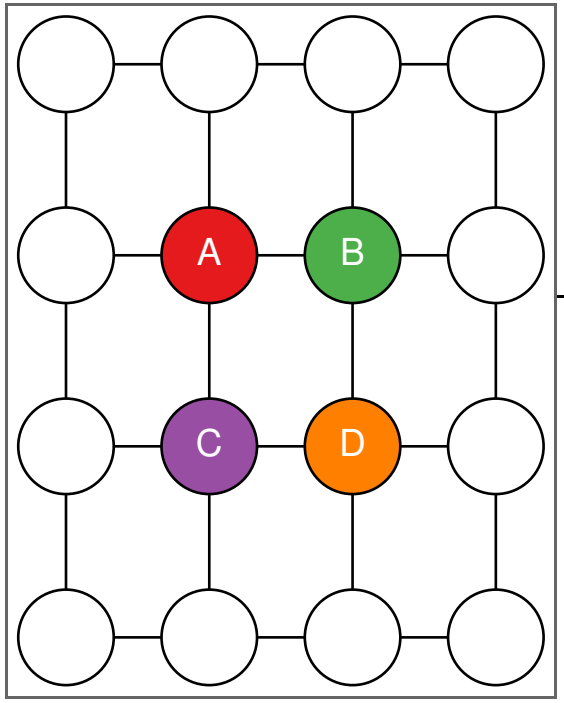
```
beta_topo_x = -CS%MEKE_topographic_beta * FatH * 0.5 *
( &
              (G%bathyT(i+1,j)-G%bathyT(i,j)) *
G%IdxCu(I,j)  &
          / max(G%bathyT(i+1,j),G%bathyT(i,j),
GV%H_subroundoff) &
      +       (G%bathyT(i,j)-G%bathyT(i-1,j)) *
G%IdxCu(I-1,j) &
          / max(G%bathyT(i,j),G%bathyT(i-1,j),
GV%H_subroundoff) )

beta_topo_y = -CS%MEKE_topographic_beta * FatH * 0.5 *
( &
              (G%bathyT(i,j+1)-G%bathyT(i,j)) *
G%IdyCv(i,J)  &
          / max(G%bathyT(i,j+1),G%bathyT(i,j),
GV%H_subroundoff) + &
              (G%bathyT(i,j)-G%bathyT(i,j-1)) *
G%IdyCv(i,J-1) &
```

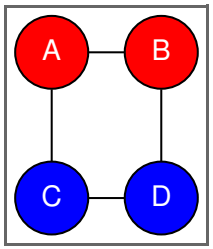
```
           / max(G%bathyT(i,j),G%bathyT(i,j-1),
GV%H_subroundoff) )
```

Index rotation ensures directional consistency

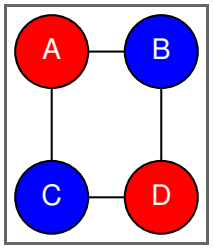
INVARIANT STENCILS

$$\phi^{(c)}_{i,j} = \frac{1}{4}(\phi_A + \phi_B + \phi_C + \phi_D)$$





$$\frac{1}{4}((\phi_A + \phi_B) + (\phi_C + \phi_D))$$



$$\frac{1}{4}((\phi_A + \phi_D) + (\phi_B + \phi_C))$$

ROTATIONAL ORDERING

When all else fails, reorder the algorithm:

```
subroutine advect_tracer(...)
   ! ...
   x_first = modulo(turns, 2) == 1
   if (x_first) then
      call advect_x(...)
      call advect_y(...)
   else
      call advect_y(...)
      call advect_x(...)
   endif
end subroutine advect_tracer
```

SUMMARY

- MOM6 test suite:
    - Verification of design requirements
        - Universal invariance rules
    - Validation of solutions
        - Site-specific regression tests
- Bit reproducibility is essential, and achievable!
- Over 40 bugs have been detected and fixed

DIMENSIONAL SCALING EXAMPLE

https://github.com/NOAA-GFDL/MOM6/pull/921

```
Kd_lay(i,j,k-1) = Kd_lay(i,j,k-1) + 0.5**KS_extra(i,K)
Kd_lay(i,j,k)   = Kd_lay(i,j,k)   + 0.5**KS_extra(i,K)
```

$\ldots + \left(\frac{1}{2}\right)^{\kappa_S}$?

ROTATIONAL EXAMPLE

https://github.com/NOAA-GFDL/MOM6/pull/1050

```
subroutine thickness_diffuse_full
   !...
   Work_u(I,j) = Work_u(I,j) + G_scale * (...)
   Work_v(i,J) = Work_v(i,J) - G_scale * (...)
   !...
end subroutine thickness_diffuse_full
```

INDEXING EXAMPLE

Assumed 1-based start index

```
subroutine register_time_deriv(...)
   real, dimension(:,:,:), target :: f_ptr
   real, dimension(:,:,:), target :: deriv_ptr
   ! ...
end subroutine register_time_deriv
```

Fails for 0-based symmetric memory grids!

ANOTHER INDEXING EXAMPLE

78d2dc3ee9a018f30bc666bd574e21fb7786403d
Extended domain to accommodate symmetric grids:

```
do J=js,je ; do i=is,ie
   h_vel = 0.5*((htot_fast(i,j) + htot_fast(i+1,j)) +
h_neglect)
   uDml_diag(I,j) = uDml_diag(I,j) / (0.01*h_vel) *
G%IdyCu(I,j) * (PSI(0.)-PSI(-.01))
enddo ; enddo
```

```
do J=js,je ; do i=is-1,ie
   h_vel = 0.5*((htot_fast(i,j) + htot_fast(i+1,j)) +
h_neglect)
   uDml_diag(I,j) = uDml_diag(I,j) / (0.01*h_vel) *
G%IdyCu(I,j) * (PSI(0.)-PSI(-.01))
enddo ; enddo
```

DEVELOPMENT GUIDELINES

1. Use parentheses!
    1. Are they honored?
    2. Am I preserving residuals?
2. Use `reproducing_sum()`
    1. Even better: Don't do global sums!
3. Assign dimensions
4. Use rotationally invariant stencils
5. Test early and often