TPLS 3.0 and Its Use of PETSc

David Scott
Edinburgh Parallel Computing Centre
Contributors

• Iain Bethune
• Toni Collis
• Lennon O’Naraigh
• David Scott
• Peter Spelt
• Prashant Valluri

Funded by EPSRC through the eCSE programme


Evaporating droplets
# History of Public Releases

<table>
<thead>
<tr>
<th>Year</th>
<th>Version</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>2013</td>
<td>1.0</td>
<td>Hand coded (generally J-SOR) solvers except for the (PETSc) pressure solver. Serial I/O.</td>
</tr>
<tr>
<td>2015</td>
<td>2.0</td>
<td>Parallel I/O (NetCDF). Introduced configuration files. 2D domain decomposition.</td>
</tr>
<tr>
<td>2017</td>
<td>3.0</td>
<td>3D domain decomposition. PETSc solvers available for the momentum calculations. Different densities for the component fluids.</td>
</tr>
</tbody>
</table>
TPLS 3.0

- Available from Sourceforge: https://sourceforge.net/projects/tpls/ under a BSD-style licence.
TPLS 3.0 – Density Contrast Flows

• Rayleigh – Taylor instability
  - Two layers of liquid with the upper being the denser.
• Stably stratified, parallel, two-phase flows
  - Two layers of fluid with the upper being the less dense.
  - The fluids are flowing in the same direction.

• Characteristics of the simulations:
  - Flows involving many length and time scales.
  - Flows with sharp changes in interfacial topologies.
  - Transient three-dimensional simulations required over long periods of time

• Require scalable code run at very high resolutions
TPLS 3.0: The Equations

Two-phase, incompressible, Navier–Stokes equations with interface capturing.

\[
\rho(\phi) \left( \frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} \right) = -\nabla p + \frac{1}{Re} \nabla \cdot \left[ \mu(\phi)(\nabla \mathbf{u} + \nabla \mathbf{u}^T) \right] + \mathbf{f}_{st}(\phi) + \rho(\phi) \mathbf{g}
\]

where \( \nabla \cdot \mathbf{u} = 0 \), \( \mathbf{g} \) is gravity and \( \phi \) is the interface capturing field.

\[
\frac{\partial \phi}{\partial t} + \mathbf{u} \cdot \nabla \phi = 0, \quad \mathbf{f}_{st} = \delta_{\epsilon}(\phi) \frac{1}{We} \hat{n} \nabla \cdot \hat{n}, \quad \hat{n} = \frac{\nabla \phi}{|\nabla \phi|}.
\]
TPLS 3.0: The Technicalities

Marker-and-cell discretisation: pressures, densities, viscosities and $\varphi$ at cell centres, velocities at cell faces.

Finite volumes, with flux-conservative differencing for the momentum equation.


Projection method: momenta are updated first, followed by a correction step involving a pressure update, thereby enforcing incompressibility.

The levelset function, $\varphi$, is carried with the flow (3rd order WENO) but is corrected at each time step (‘redistancing’).
• Heavy fluid sitting on top of a light fluid (gravity acting downwards)
• System starts from rest with a sinusoidal perturbation to the interface
• Heavy fluid accelerates downwards forming complicated interfacial structures due to rollup of vorticity
• System parametrised by the Atwood number

\[ At = \frac{\rho_{\text{heavy}} - \rho_{\text{light}}}{\rho_{\text{heavy}} + \rho_{\text{light}}}, \]

• In simulations, \( \rho_{\text{heavy}} = 3 \rho_{\text{light}} \)
TPLS 3.0: Rayleigh-Taylor Instability

TPLS 3.0: Stratified Flows

• A pressure-driven channel flow with two stably stratified phases

• At high flow rates (Reynolds numbers), an unstable equilibrium sets in

• Interfacial waves develop and can evolve to breaking waves, ligaments, billows, droplets etc.

• The density ratio, $r$ (bottom phase vs top phase) is a key parameter
TPLS 3.0: Stratified Flows

Interface shape evolution from DNS for a case at moderate density ratio, $r = 10$. Other parameters: $(m = 50, h_0 = 0.2, G = 0.1, We = 10)$, and $Re = 500$. 
How to Use TPLS

• Describe the initial configuration
  • initial_config.opt

• Generate the initial configuration
  • run ./create_initial_configuration

• Configure how the program works (non-PETSc options)
  • tpls_config.opt

• PETSc run-time configuration
  • .petscrc
## initial_config.opt

- **## Domain grid**
  - # Number of grid points in X (l), Y (m) and Z (n) directions.
  - maxl 257
  - maxm 145
  - maxn 153

- **## Interface detection method.**
  - # Options:
  - # lsm - level-set method (default)
  - # dim - diffuse interface method
  - idm lsm

- **## Flow type.**
  - # Options:
  - # channel - channel flow
  - # rti - Rayleigh Taylor Instability
  - flow_type channel
initial_config.opt (cont.)

- ## Fluid flow

- # Reynolds number.
  - re 1.0

- # Viscosity and density of the lower fluid.
  - mu_minus 1.0
  - rho_minus 1.0

- # Viscosity and density of the upper fluid.
  - mu_plus 1.0
  - rho_plus 3.0

- # Interface height, or height of lower liquid layer, expressed as
  - # a proportion where 0 <= height <= 1.
  - height 0.5
initial_config.opt (cont.)

• # Pressure gradient.
  • dpdl -1.0

• # Gravity.
  • Grav 1.0
  • gz -1.0

• # Surface tension scaling parameter.
  • scap 0.01

• # Time step (>0).
  • dt 0.0001

• # Smooth width scale factor.
  • smooth_width_scale 1.5
### Process grid

Number of processes in the X, Y and Z dimensions, which defines the process grid which overlays the domain grid.

The following conditions need to be respected:

- Number of processes available = num_procs_x * num_procs_y * num_procs_z
- num_procs_x is a divisor of (maxl - 1)
- num_procs_y is a divisor of (maxm - 1)
- num_procs_z is a divisor of (maxn - 1)
- num_procs_x >= 1
- num_procs_y >= 1
- num_procs_z >= 1

TPLS will raise an error if these conditions do not hold.

num_procs_x 16
num_procs_y 6
num_procs_z 8
### Selection of PETSc or original equation solvers.
- `#` T or F for PETSc or original.
- `petsc_solver_u` T
- `petsc_solver_v` T
- `petsc_solver_w` T
- `petsc_solver_p` T

### Is a solver to be monitored?
- `#` T or F.
- `u_monitoring_on` F
- `v_monitoring_on` F
- `w_monitoring_on` F
- `p_monitoring_on` F
## tpls_config.opt (cont)

- ## Original momentum equation solver configuration
- # Number of iterations in solvers for u, v and w velocities (>= 1).
  - mom_u 30
  - mom_v 30
  - mom_w 30
- ## Level-set equation solver configuration
- # Number of iterations in solver (>= 1).
  - levelset 10

- # Maxu
  - maxu 10.0
• ## TPLS operation
• # PHI channel .dat file output frequency (>= 1).
  • phi_dat_frequency 1000
• # UVW channel .dat file output frequency (>= 1).
  • uvw_dat_frequency 1000
• # Backup channel .dat file output frequency (>= 1).
  • backup_frequency 1000
• # Backup files in netCDF hdf5 format (= T or F).
  • backup_hdf5_format T
• # Number of timesteps (>= 1).
  • num_timesteps 1000

• ## DIM equation solver configuration
• # Number of iterations
  • max_iteration_dim 18
Run-Time Configuration of PETSc
.petscrc

• -u_ksp_rtol 0.00000007
• -u_ksp_final_residual
• -v_ksp_rtol 0.000001
• -v_ksp_final_residual
• -w_ksp_rtol 0.0000002
• -w_ksp_final_residual
• -p_ksp_rtol 0.000009
• -p_ksp_type minres
• -p_pc_type sor
• -p_pc_sor_omega 1.5
• -p_ksp_final_residual

• Note that one can configure the PETSc (Krylov) solvers individually through a prefix to the configuration options. For example the generic option ksp_rtol becomes u_ksp_rtol, etc. As we shall see, the prefixes are specified in the code.
Why Use PETSc

• PETSc provides support for structured grids.

• call DMDACreate3d(PETSC_COMM_WORLD, &
  DM_BOUNDARY_PERIODIC, &
  DM_BOUNDARY_PERIODIC, &
  DM_BOUNDARY_NONE, &
  DMDA_STENCIL_BOX, &
  global_dim_x, global_dim_y, global_dim_z+2, &
  num_procs_x, num_procs_y, num_procs_z, &
  dof, stencil_width,
  petsc_x_lengths,
  PETSC_NULL_INTEGER, &
  petsc_z_lengths_p, &
  da_s, ierr)

• This creates da_s which may then be used to create PETSc vectors.
Why Use PETSc (cont.)

- Two sorts of array may be created: global, distributed arrays and local arrays.
  
  - call DMGetGlobalVector(da_s, pres_vec, ierr)
  - call DMGetLocalVector(da_s, pres_lvec, ierr)

- A local vector includes room for the appropriate ghost (halo) points.

- It is simple to populate a local vector (including its halo points) given a global vector.

- call DMGlobalToLocalBegin(da_s, pres_vec, INSERT_VALUES, pres_lvec, ierr)
  - call DMGlobalToLocalEnd(da_s, pres_vec, INSERT_VALUES, pres_lvec, ierr)

- Note the absence of MPI calls. The movement of data is done behind the scenes by PETSc.
Why Use PETSc (cont.)

- Distributed matrices may also be created from DMs and used in conjunction with KSPs to solve linear systems.

```plaintext
- call DMCreateMatrix(da_s, A, ierr)
- call compute_rhs_p(ksp_pres, b_vec, ierr)
- call compute_matrix_p(ksp_pres, A, A, ierr)
- call KSPSetOperators(ksp_pres, A, A, ierr)
- call KSPSolve(ksp_pres, b_vec, pres_vec, ierr)
- call MatDestroy(A, ierr)
```

This requires that a relationship has been established between the DM and the KSP.
Why Use PETSc (cont.)

- KSPs provide access to PETSc linear solvers. A KSP may be associated with a DM as illustrated here.

  - call KSPCreate(PETSC_COMM_WORLD, ksp_u, ierr)
  - call KSPSetOptionsPrefix(ksp_u, 'u_', ierr)
  - call KSPSetFromOptions(ksp_u, ierr)
  - call KSPSetComputeInitialGuess(ksp_u, set_initial_guess_u, PETSC_NULL_OBJECT, ierr)
  - call KSPSetDM(ksp_u, da_u, ierr)
  - call KSPSetDMActive(ksp_u, PETSC_FALSE, ierr)

- One can see that a prefix has been associated with the KSP so that it can have its own run-time, configuration options.
TPLS 3.0 Performance

• 3D decomposition accelerates speed (2 x TPLS2.0 on 1,536 cores)
• Krylov solvers do not lead to an increase in performance but have different termination criteria
• Density contrast doubles the execution time
• Refer to the eCSE report for full performance analysis
Further Work

• Merge existing code on counter-current flows and droplet formation.

• Include heat transfer, mass transfer with reaction and interfacial phase change.

• Implement complex geometries.