Test (2): GFD Jet in a periodic channel

Purpose

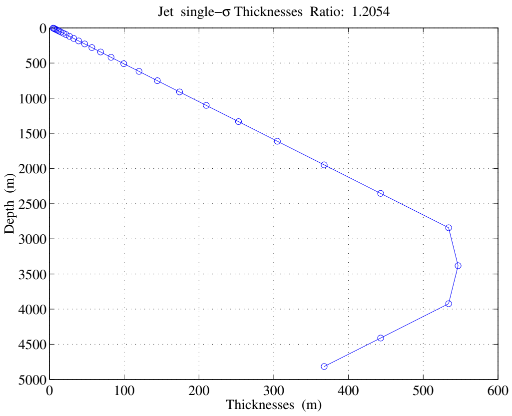
To demonstrate the consistency between the rigid–lid streamfunction, rigid–lid surface pressure and free surface (surface pressure) formulations in a controlled environment.

Status

A simple Cartesian channel domain is defined as

$$\begin{aligned} n_x,\ n_y &= 100,\ 100 \\ dx,\ dy &= 10\text{km},\ 10\text{km} \\ lat_c,\ lon_c &= 45\text{N},\ 30\text{W} \end{aligned}$$

30 model levels are spread throughout the water column. The first is 5m thick. Subsequent levels are increasingly thicker by a factor of ∼1.2 until the bottom–most levels which are "folded" to enhance bottom resolution. The level distribution is pictured to the right.



The bottom is flat at 5000m. The north and south walls are straight land masks (4pt and 2pt thick respectively). The resulting channel is periodic in the east–west direction with a 2 grid-point overlap to provide 2$^{\text{nd}}$ order accurate continuity in the first derivatives.

A single period of a sinusoidal jet is placed in the center of the domain. The amplitude of the wave is chosen to create an initially quiescent field near the solid boundaries. The mass field for the jet is that of a smooth Gulf Stream cross section patched to "flat" exterior fields. The velocity field is created by integrating up from the bottom. These fields are illustrated in figure 2.

(A) Streamfunction formulation

The streamfunction formulation simulations form the baseline for comparing the surface pressure formulation simulations. This series starts with the basic code. Various CPP options are then turned on to include the various supporting changes for the surface pressure formulation

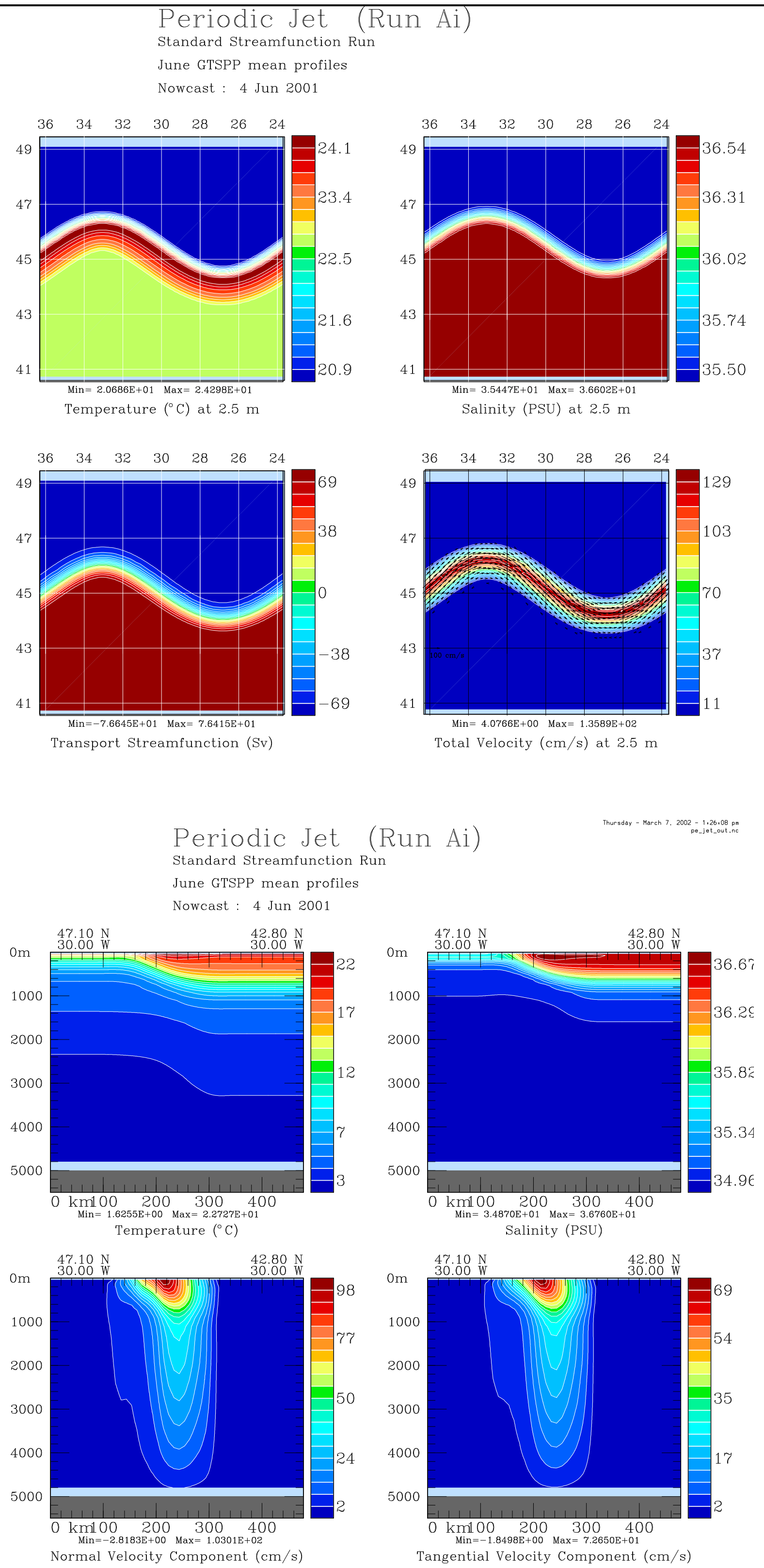


Figure 2 : Periodic channel initial jet.

(i) Baseline simulation

The baseline simulation is briefly presented in figures 3–6. The kinetic energy remains bounded and fairly constant for this test. The sinusoidal front largely remains stationary, the northern excursion shows a slight eastward translation over the course of the simulation. The surface warm core of the stream shows some erosion over the course of the simulation.

Overall the simulation is judged boring, but useful for debugging.

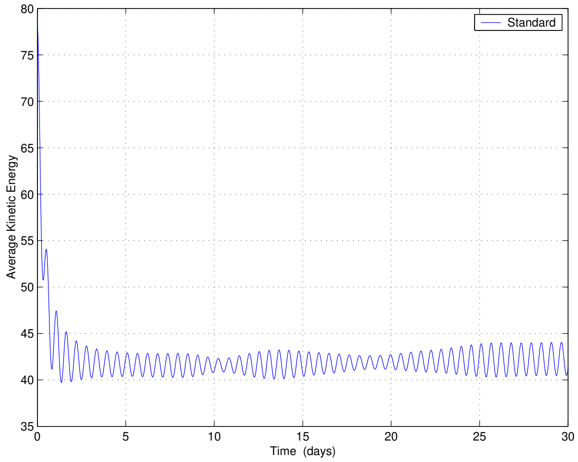


Figure 3 : Kinetic Energy for baseline simulation.

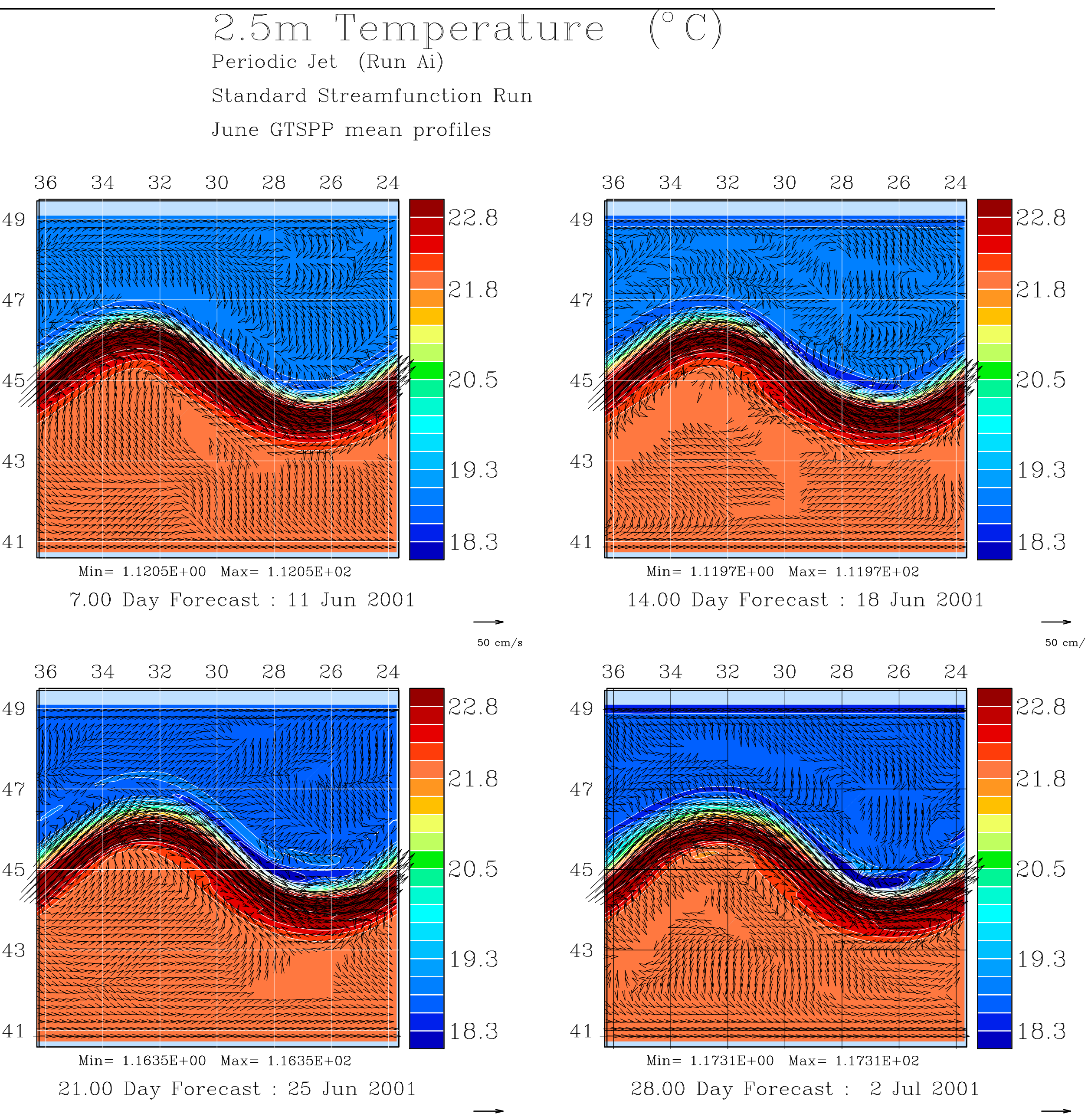


Figure 4 : Baseline streamfunction formulation. Near surface temperature with total velocity vectors.

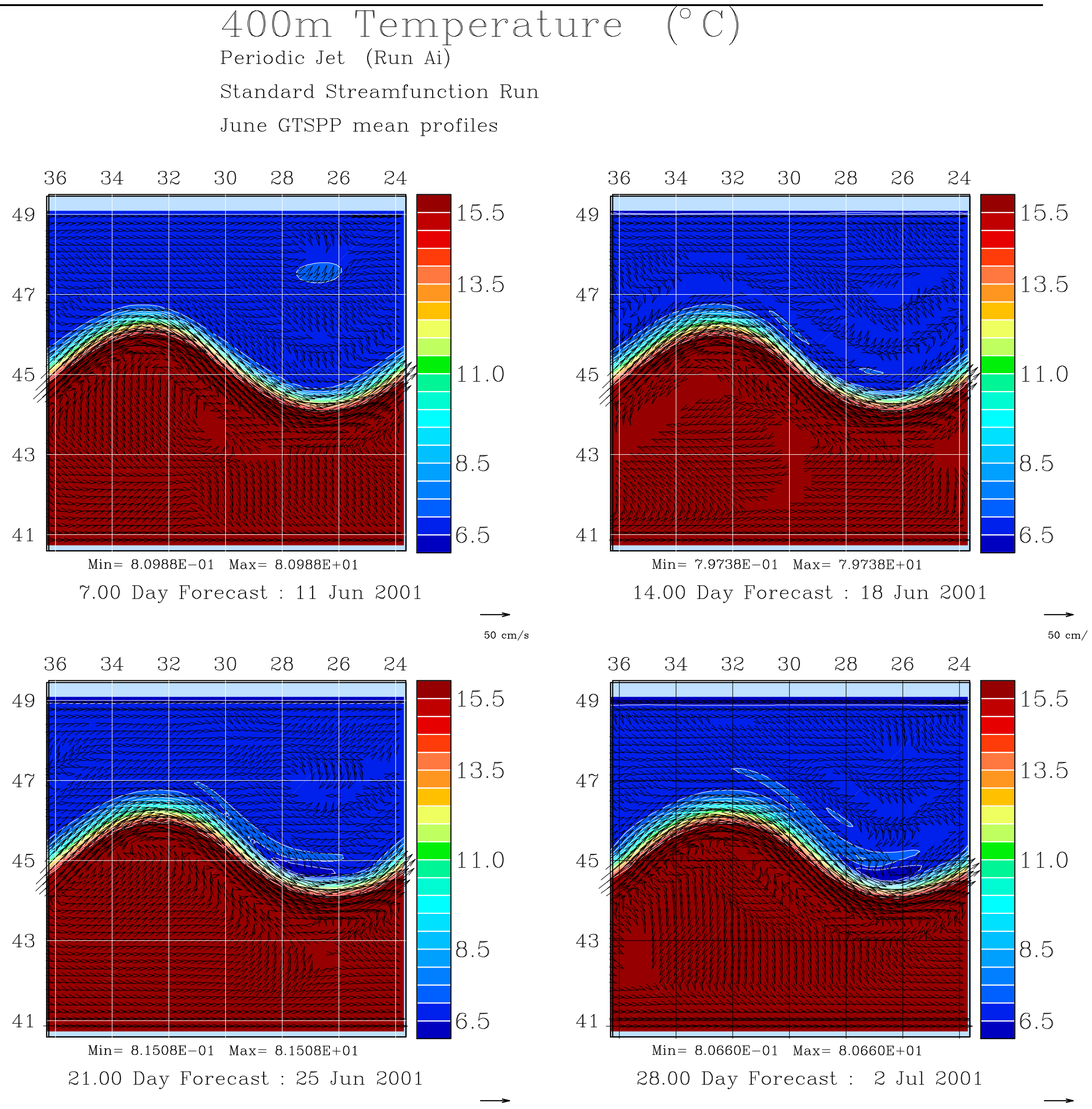


Figure 5 : Baseline streamfunction formulation. Mid–level temperature with total velocity vectors.

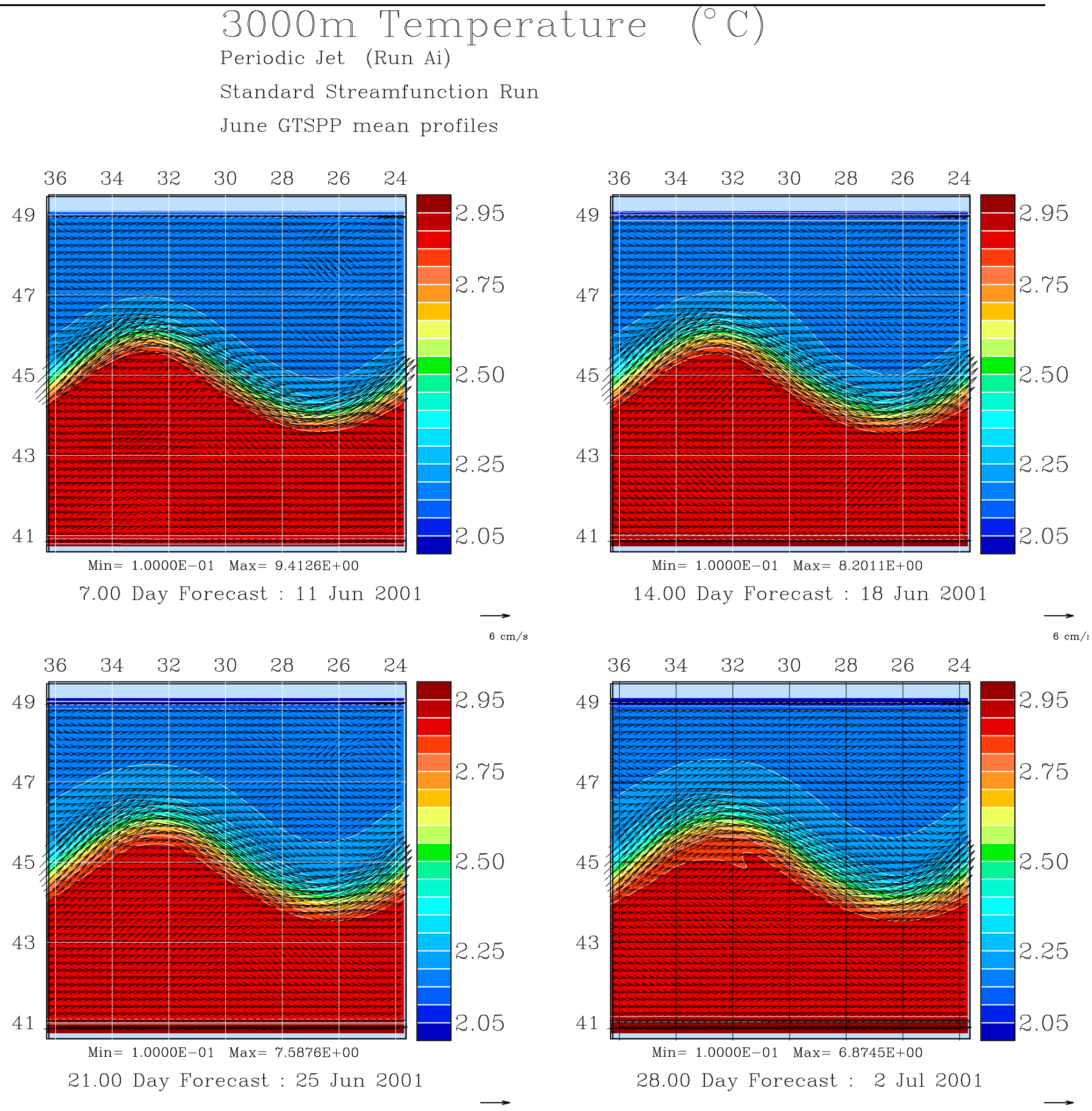


Figure 6 : Baseline streamfunction formulation. Deep temperature with total velocity vectors.

(ii) Pressure formulation momentum fluxes. (CPP option `mfluxavg`)

In order to maintain the vertically integrated conservation of mass, the pressure formulation requires a modified discretization (6 point) of the momentum fluxes. For compatibility, Lozano & Lanerolle provided the option of using this discretization in the streamfunction formulation.

A comparison between *(i)* and *(ii)* is presented in figures 7–14. The overall picture is that the modified momentum fluxes produce only minor changes. The kinetic energy oscillations remain in phase, with the KE in *(ii)* shifted up by $\sim \frac{1}{8}$ of a unit. The RMS differences remain small and the PCC stays close to 1 in all fields. The temperature difference fields show small differences, concentrated around the front.

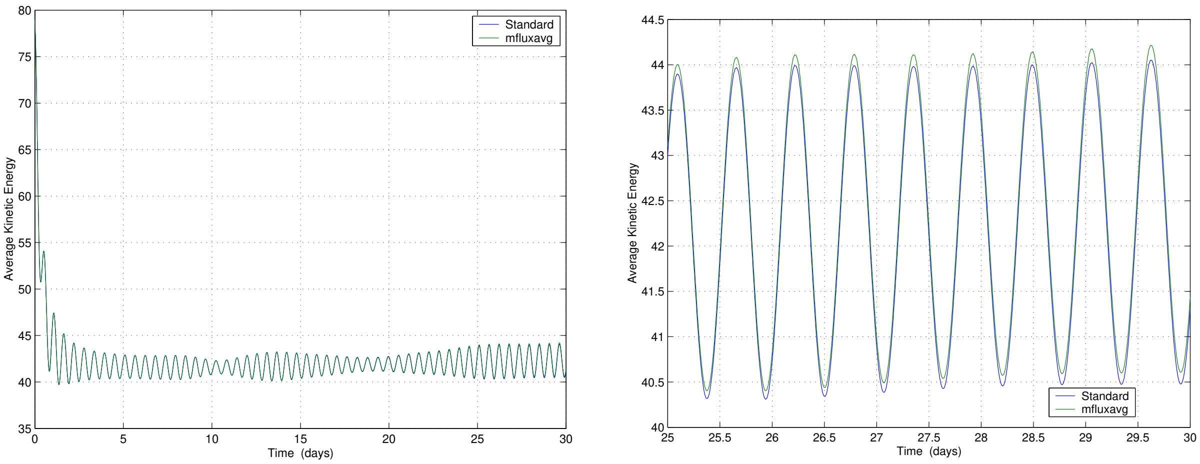


Figure 7 : Kinetic Energy comparison between baseline simulation *(i)* versus 6 point flux algorithm *(ii)* (CPP option `mfluxavg`).

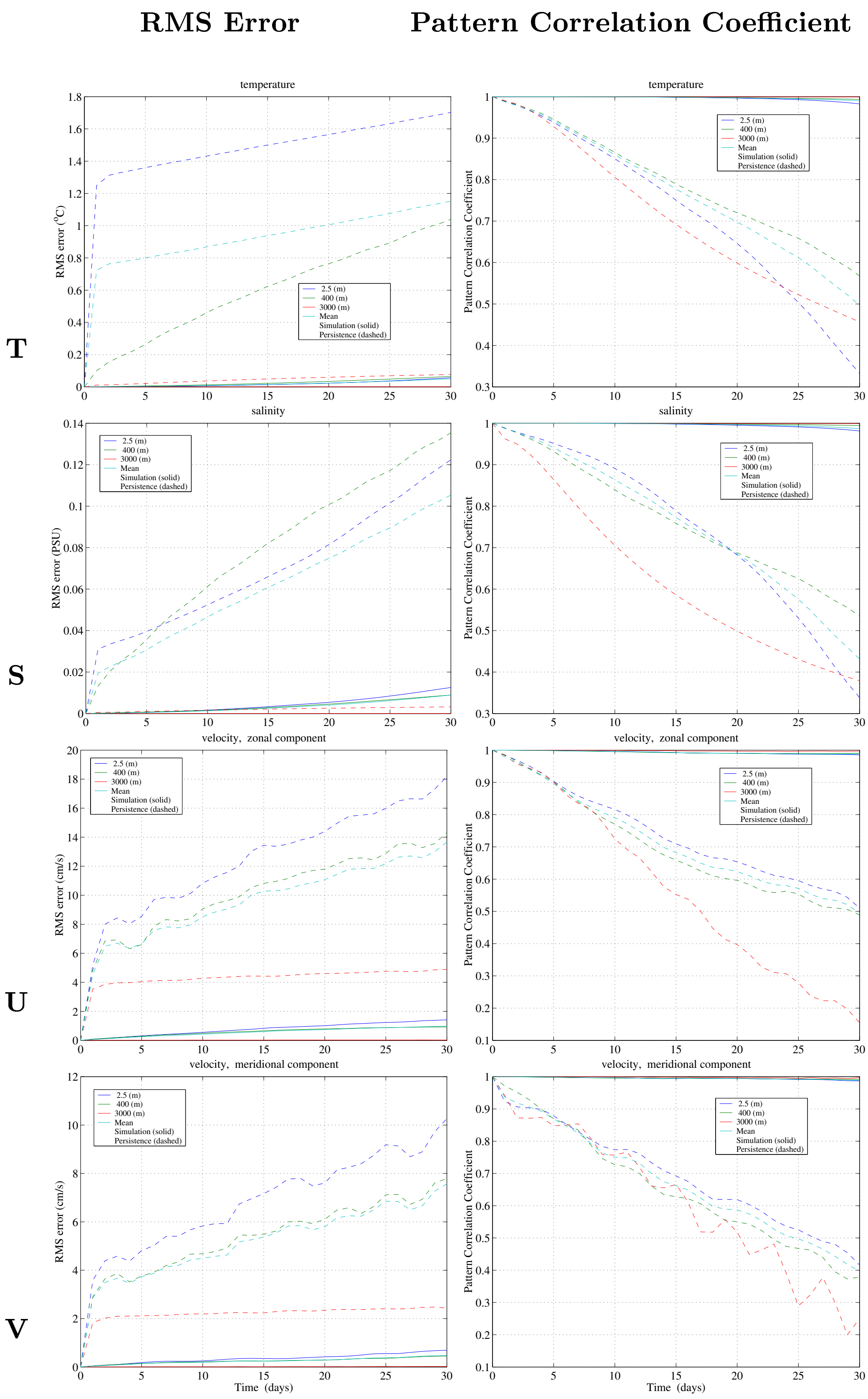


Figure 8: Comparison of baseline run *(i)* versus 6 point flux algorithm *(ii)*

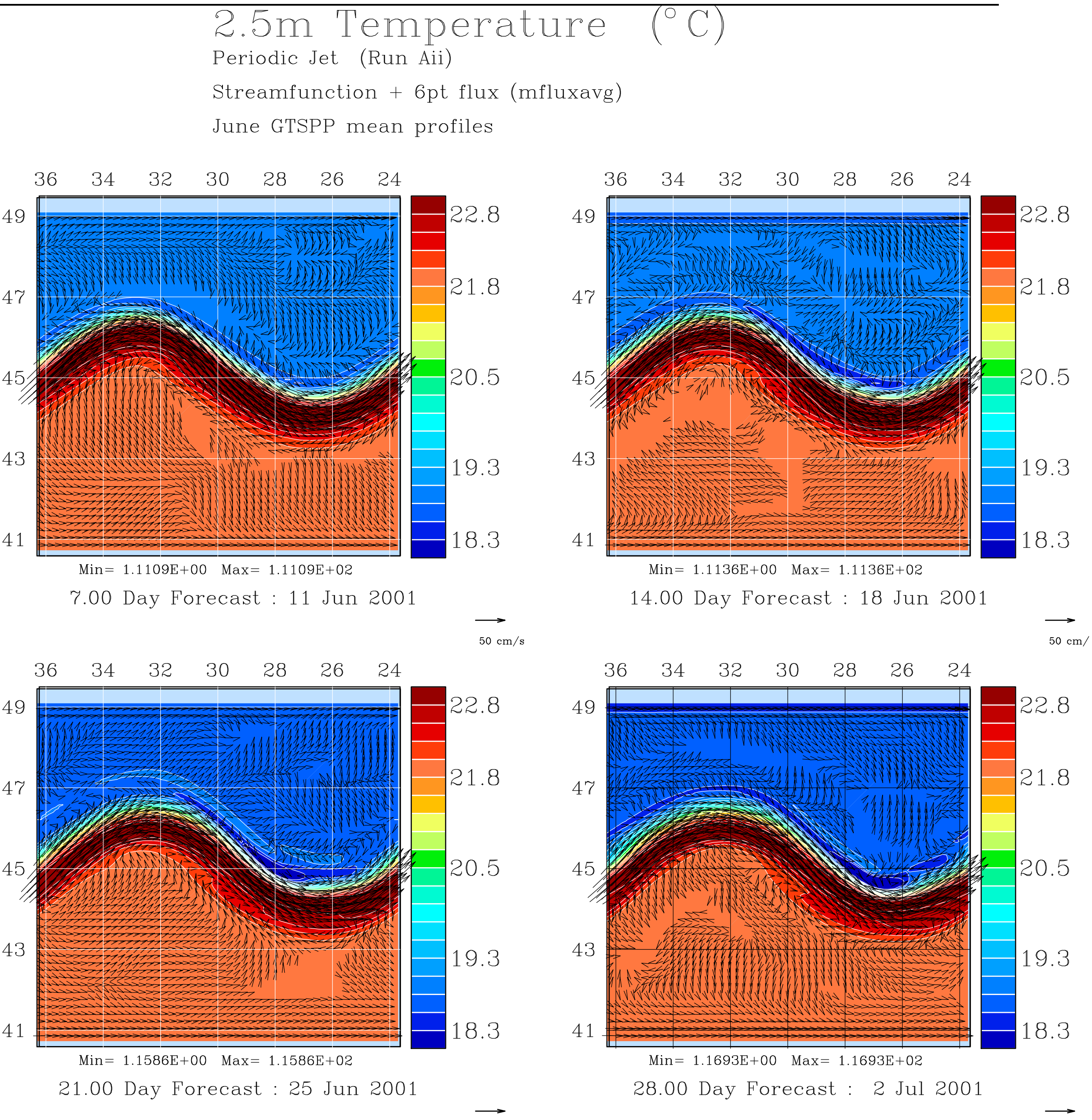


Figure 9 : Streamfunction formulation with divergence operator from surface pressure formulation. Near surface temperature with total velocity vectors.

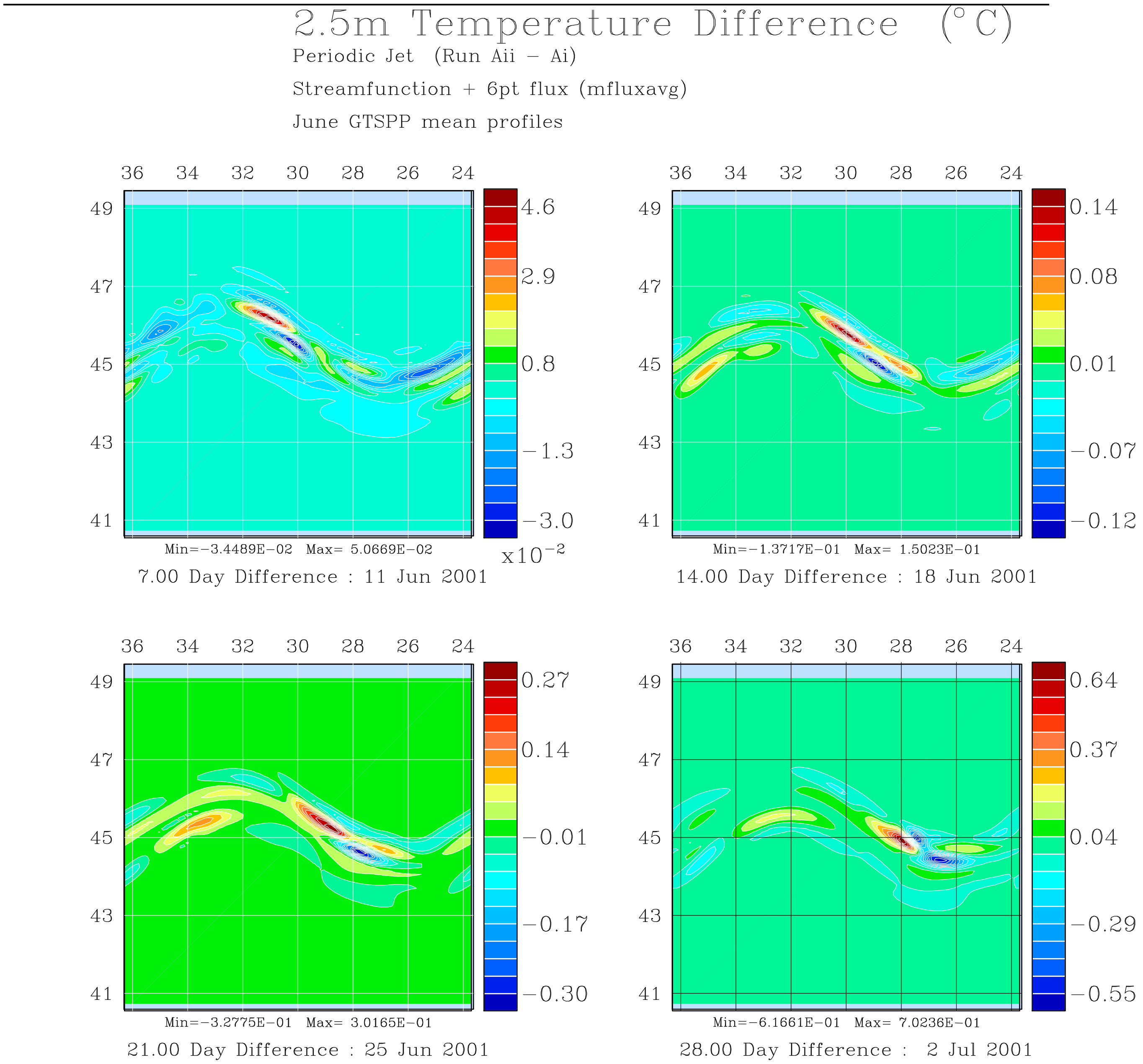


Figure 10 : Near Surface comparison of baseline simulation to simulation with 6 point flux algorithm (CPP option `mfluxavg`).

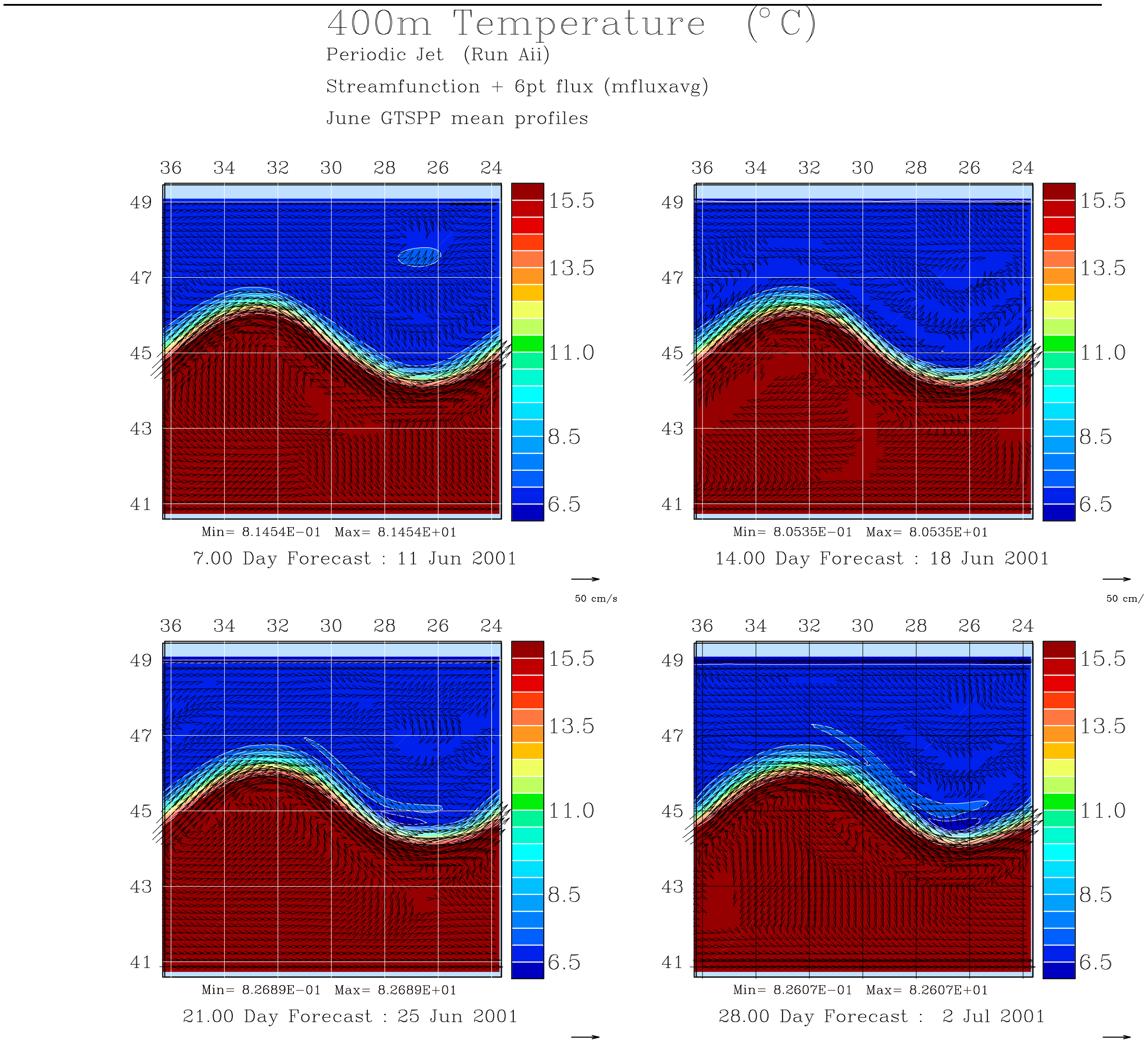


Figure 11 : Streamfunction formulation with divergence operator from surface pressure formulation. Mid–level temperature with total velocity vectors.

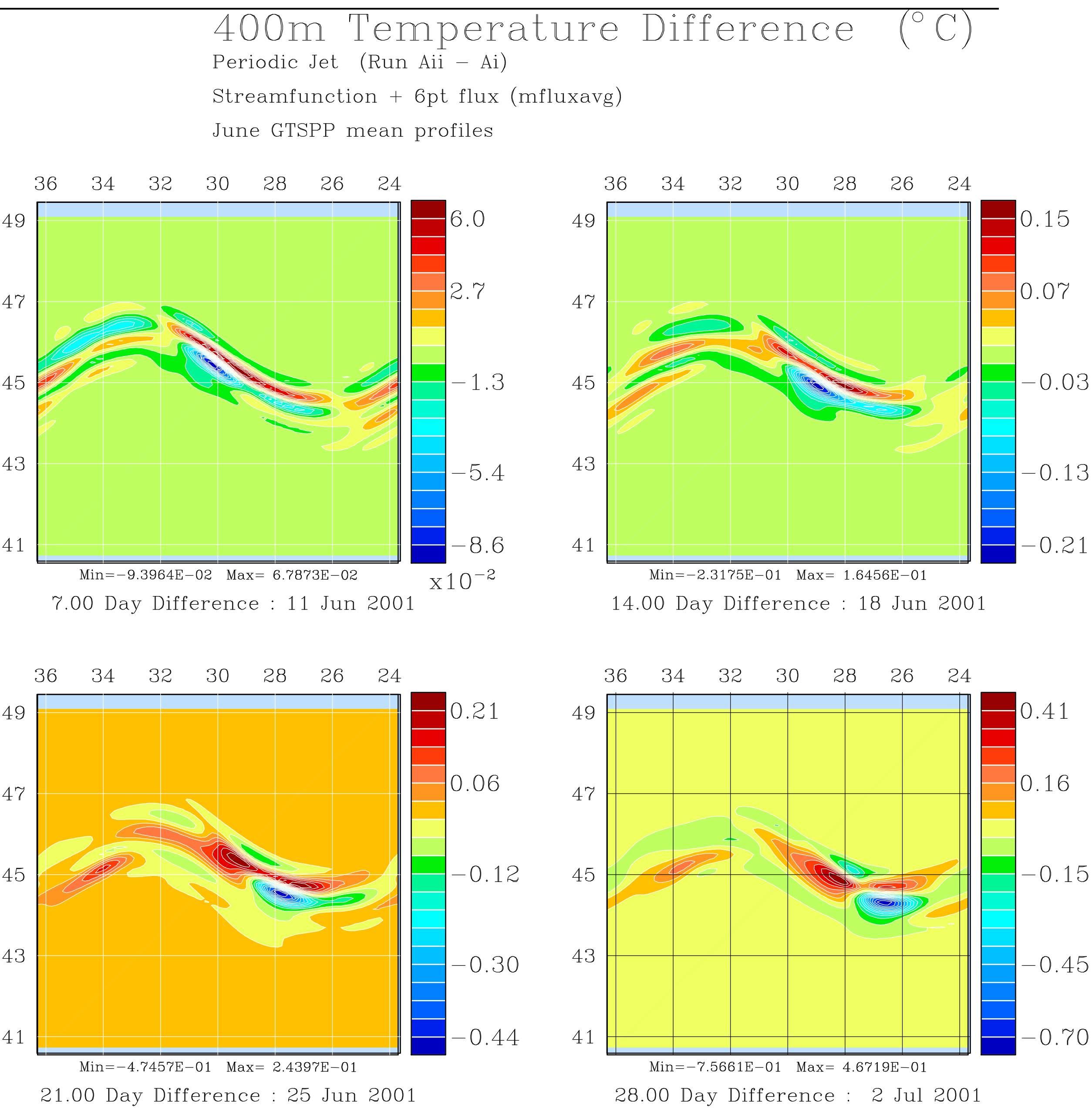


Figure 12 : Mid–level comparison of baseline simulation to simulation with 6 point flux algorithm (CPP option `mfluxavg`).

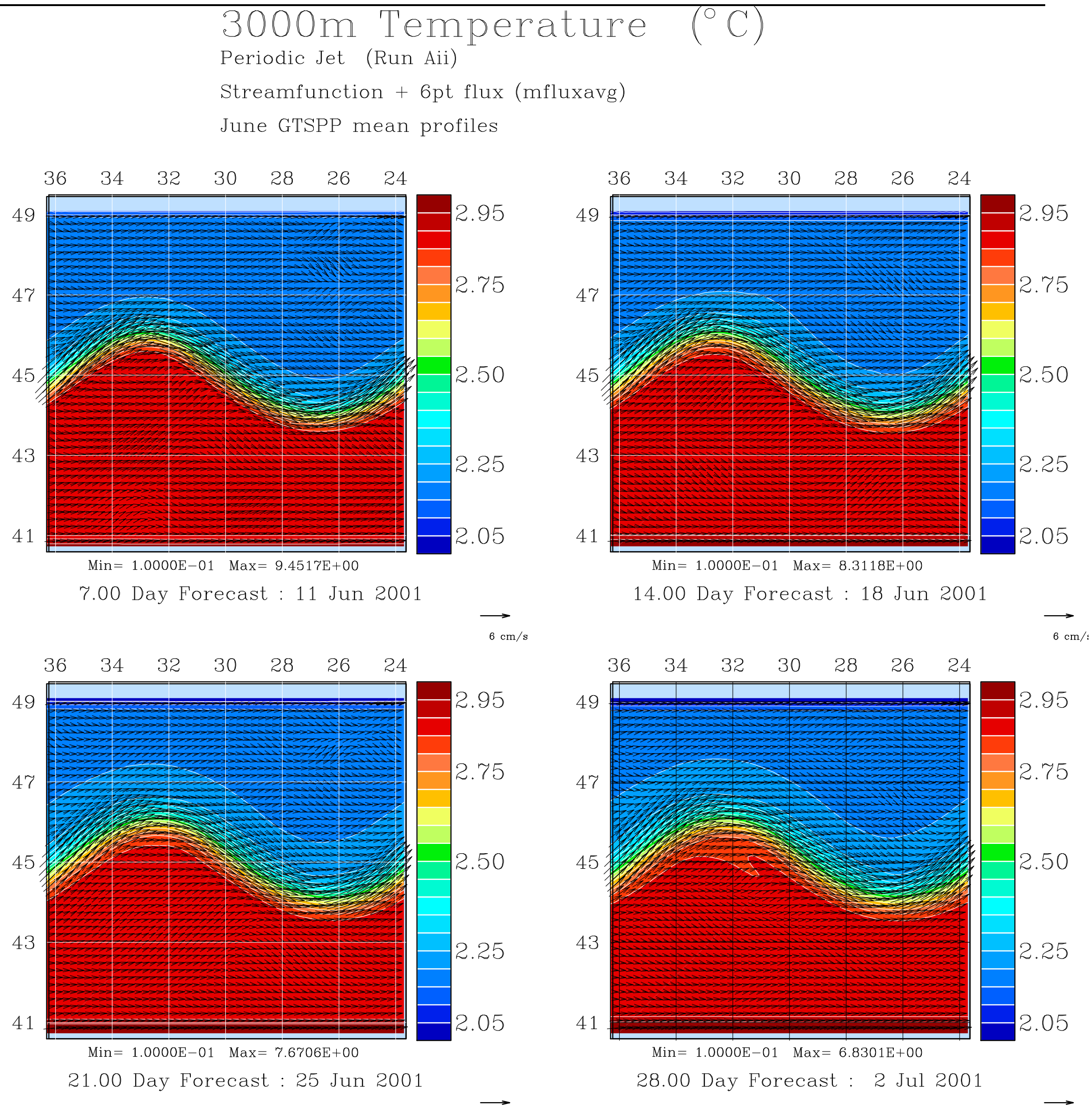


Figure 13 : Streamfunction formulation with divergence operator from surface pressure formulation. Deep temperature with total velocity vectors.

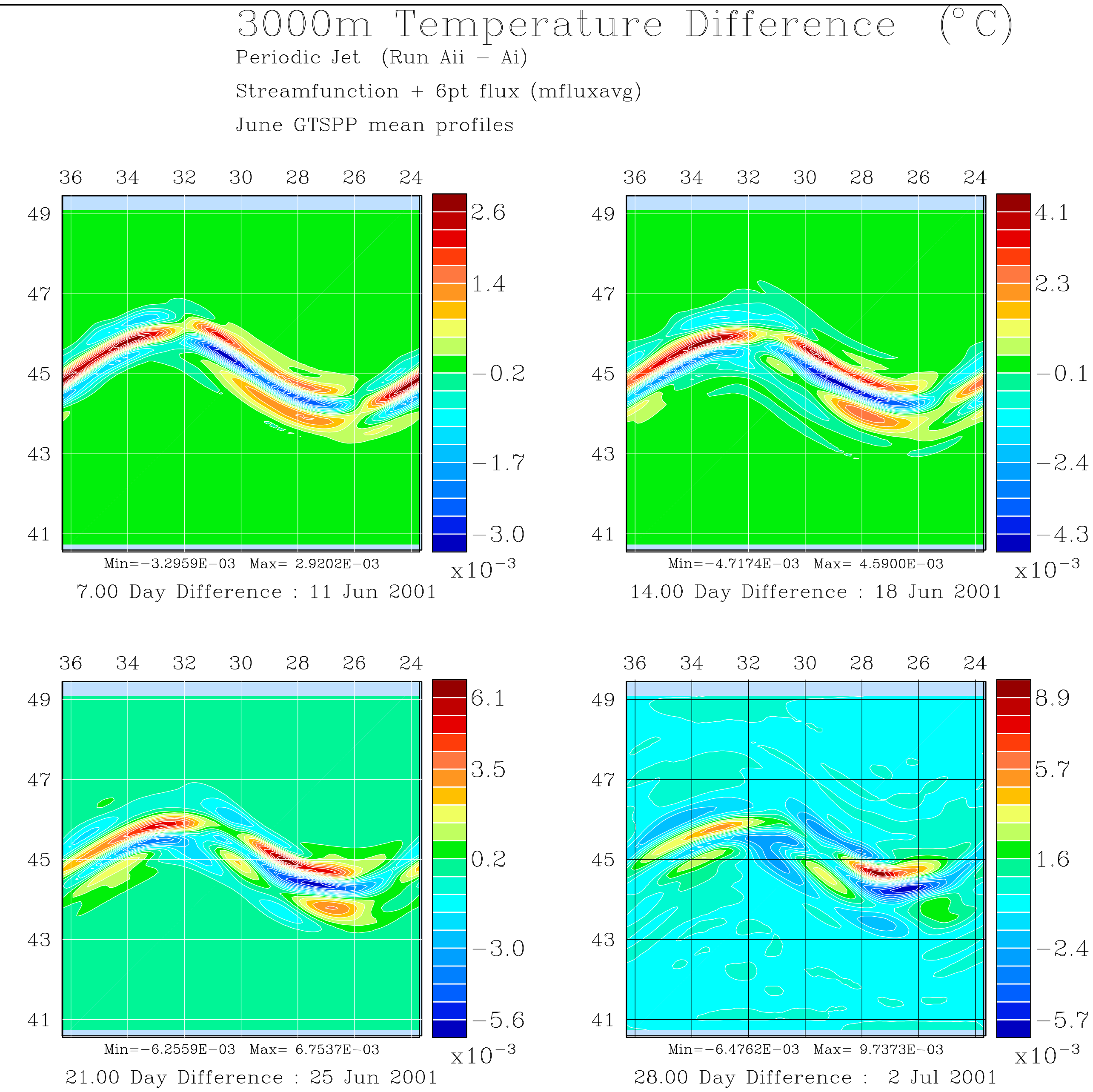


Figure 14 : Deep comparison of baseline simulation to simulation with 6 point flux algorithm (CPP option `mfluxavg`).

(iii) Pressure formulation momentum fluxes, alternative formulation. (CPP options `mflux-avg` + `tfluxes`)

The alternative formulation is a variation in the handling of land masks, specifically the method of preventing the 6–point flux operator from using invalid data under the land mask.

- In the original formulation:
  - In the absence of land masks the operator is coded without reference to masks.
  - In the presence of land masks, a binary [0,1] mask is tested before each evaluation of the velocity.
- In the new formulation, each evaluation of the velocity is multiplied by the binary mask.

In light of the fact that the periodic channel has small & fairly trivial land masks, the results are unsurprising.

- The difference fields between cases *(ii)* and *(iii)* are identically zero after 30 days.
- The timing is, unfortunately, still an open question. The two simulation were run on separate machines and even portions of *(iii)* unaffected by the CPP option `tfluxes` ran faster.

Since the results are identical to those of case *(ii)*, no figures are presented for this case.

(iv) Addition of Asselin filter. (CPP options **mfluxavg** + **AsselinFilt**)

To inhibit the odd–even decoupling of the leap–frog time–stepping, the HOPS PE model has a pair of options

(1) Intermittent reversion to forward Euler time–stepping (a first order algorithm).
(2) An Asselin time filter at every time step.

Lozano disabled the Euler time–stepping option when the surface pressure formulation is activated since:

(i) At an Euler step there is the need to add additional cumbersome code to accommodate the difference between Euler and leap frog time steps.
(ii) The Asselin filter works well.
(iii) The effects of Asselin filter versus Euler time–stepping can be explored in the streamfunction formulation.

A comparison between *(i)* and *(iv)* is presented in figures 15–22. The addition of the Asselin filter has a greater effect than modifying the momentum fluxes. The amplitudes of the KE oscillations are reduced and the oscillations themselves lag those of cases *(i–iii)* by ∼70 minutes. The Asselin filter parameter, ν, is set to 0.001 for the remaining figures by reason of its slightly larger KE oscillation (similarity to *i–iii*). The PCC statistics remain near 1, although not so close as in *ii*. The RMS differences remain small, although the meridional component of velocity is showing larger differences than *ii*. The temperature difference fields are comparable to *(ii)*.

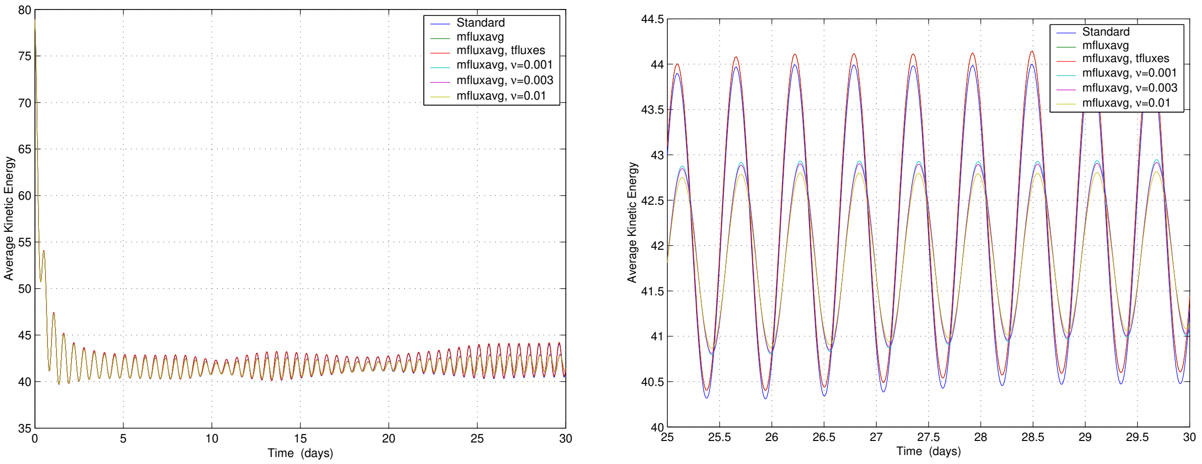


Figure 15 : Kinetic Energy comparison between baseline simulation *(i)* versus 6 point flux algorithm & Asselin filter *(iv)* (CPP options **mfluxavg** + **AsselinFilt**).

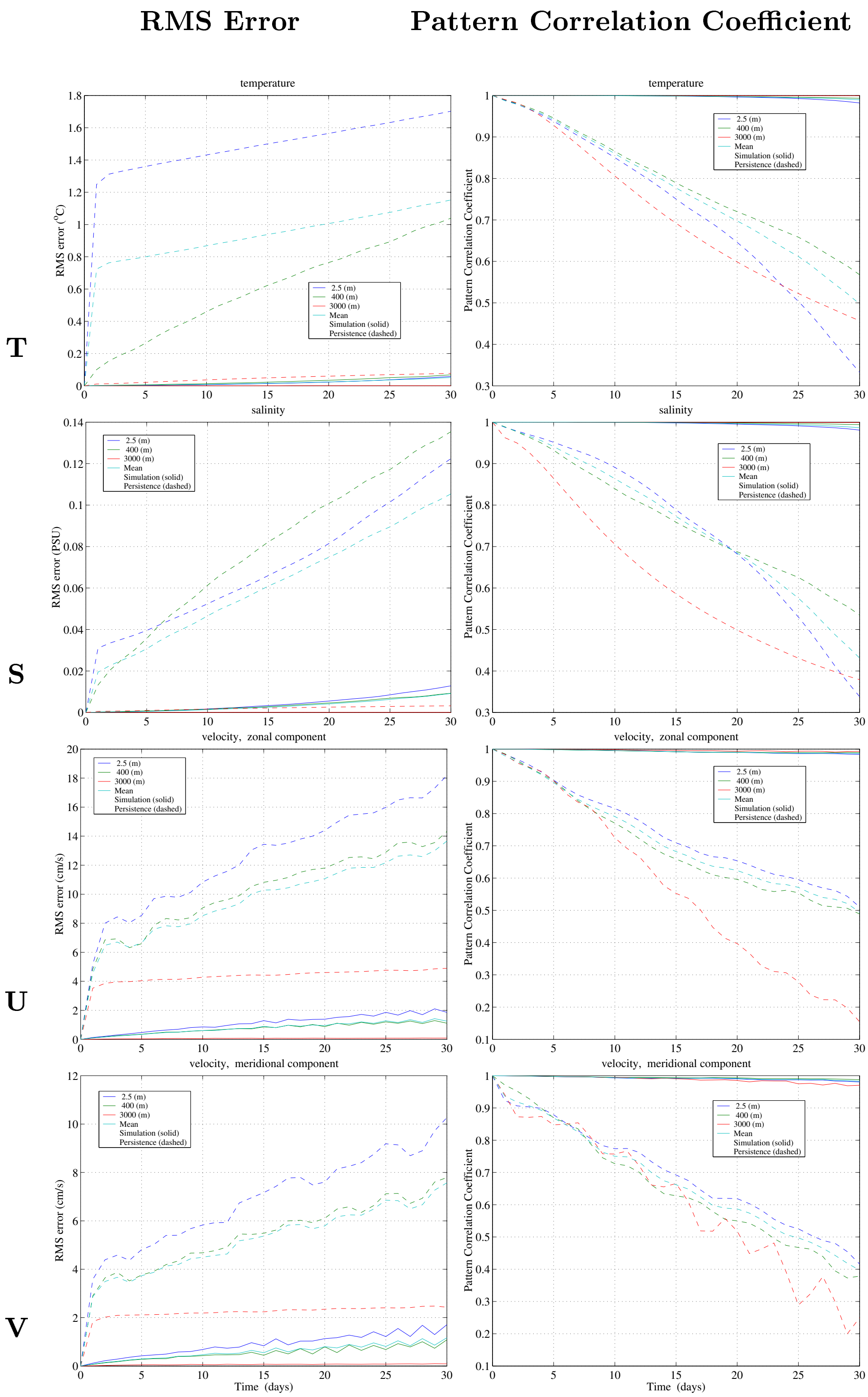


Figure 16: Comparison of baseline run *(i)* versus 6 point flux algorithm + Asselin Filter *(iv)*

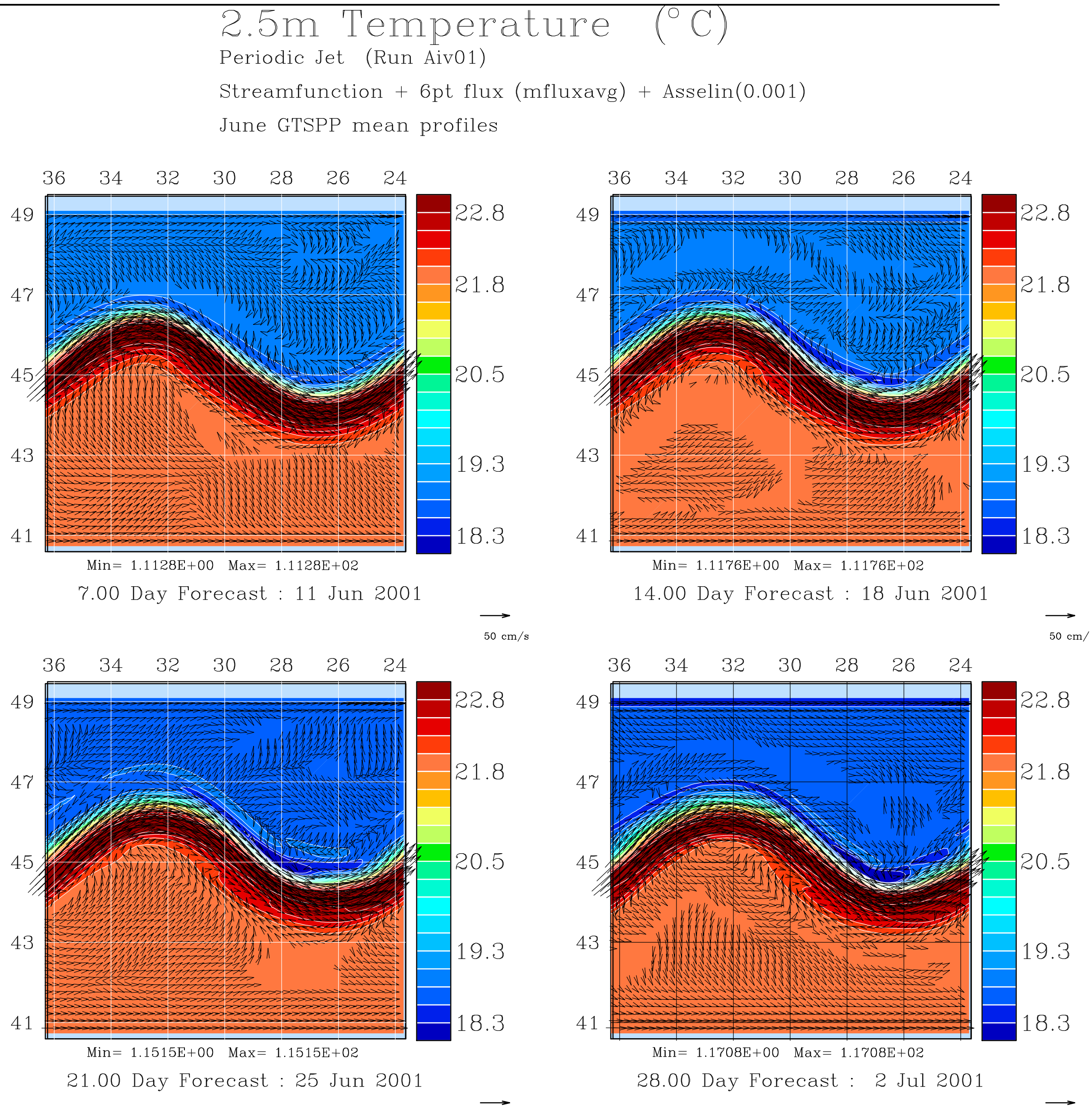


Figure 17 : Streamfunction formulation with divergence operator from surface pressure formulation plus Asselin filter. Near surface temperature with total velocity vectors.

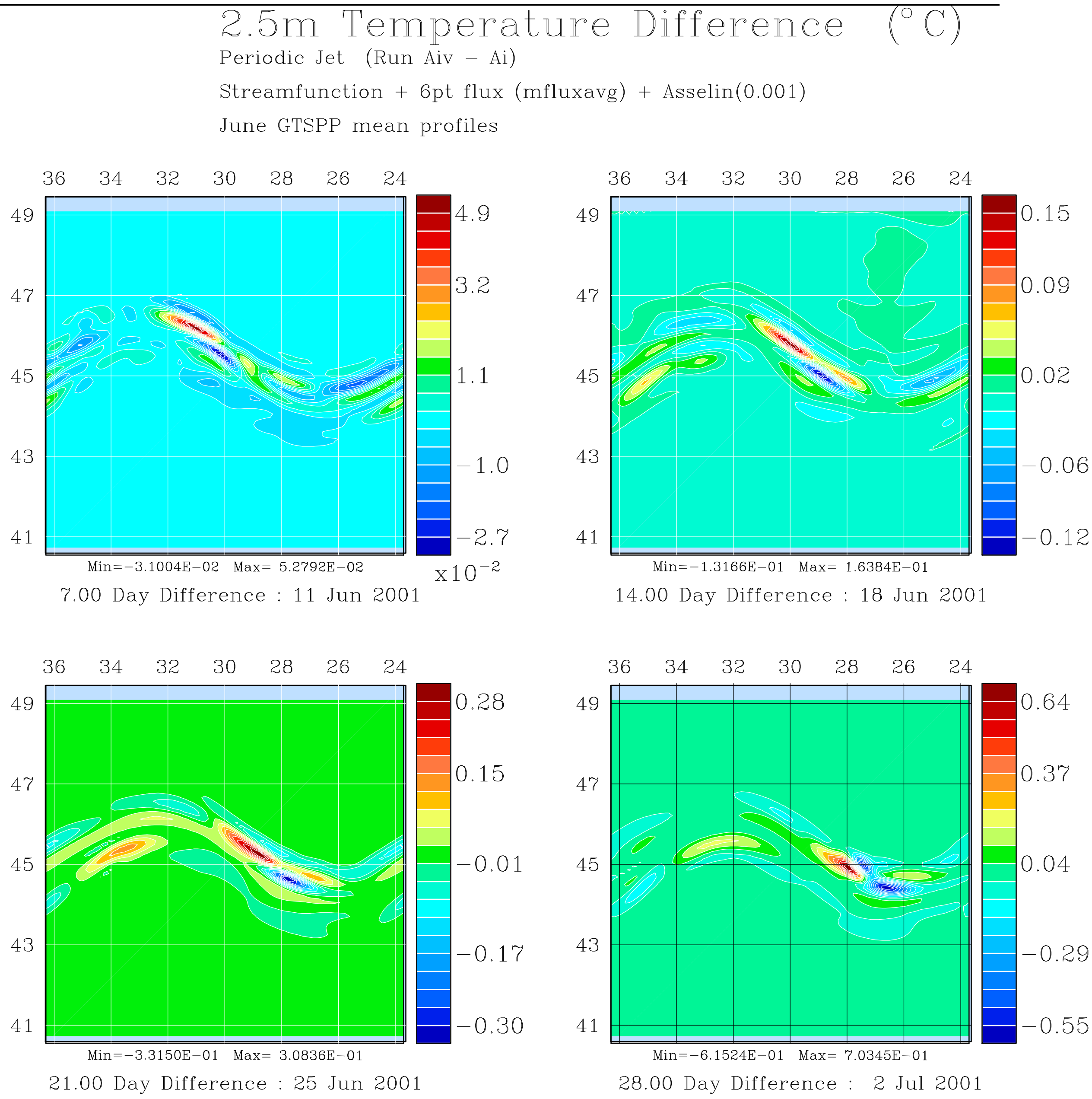


Figure 18 : Near Surface comparison of baseline simulation to simulation with divergence operator from surface pressure formulation plus Asselin filter (CPP options `mfluxavg` + `AsselinFilt`).

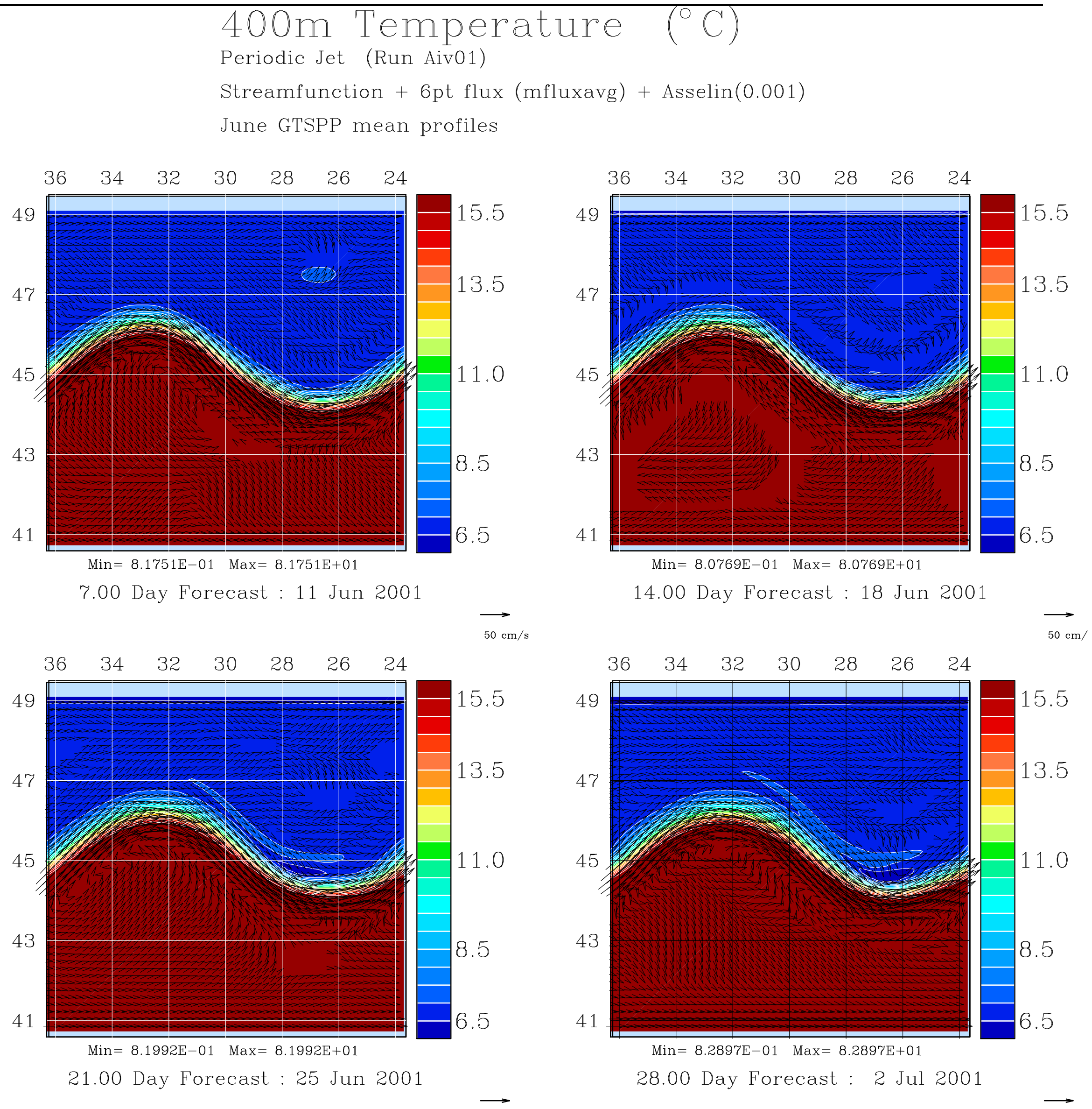


Figure 19 : Streamfunction formulation with divergence operator from surface pressure formulation plus Asselin filter. Mid–level temperature with total velocity vectors.

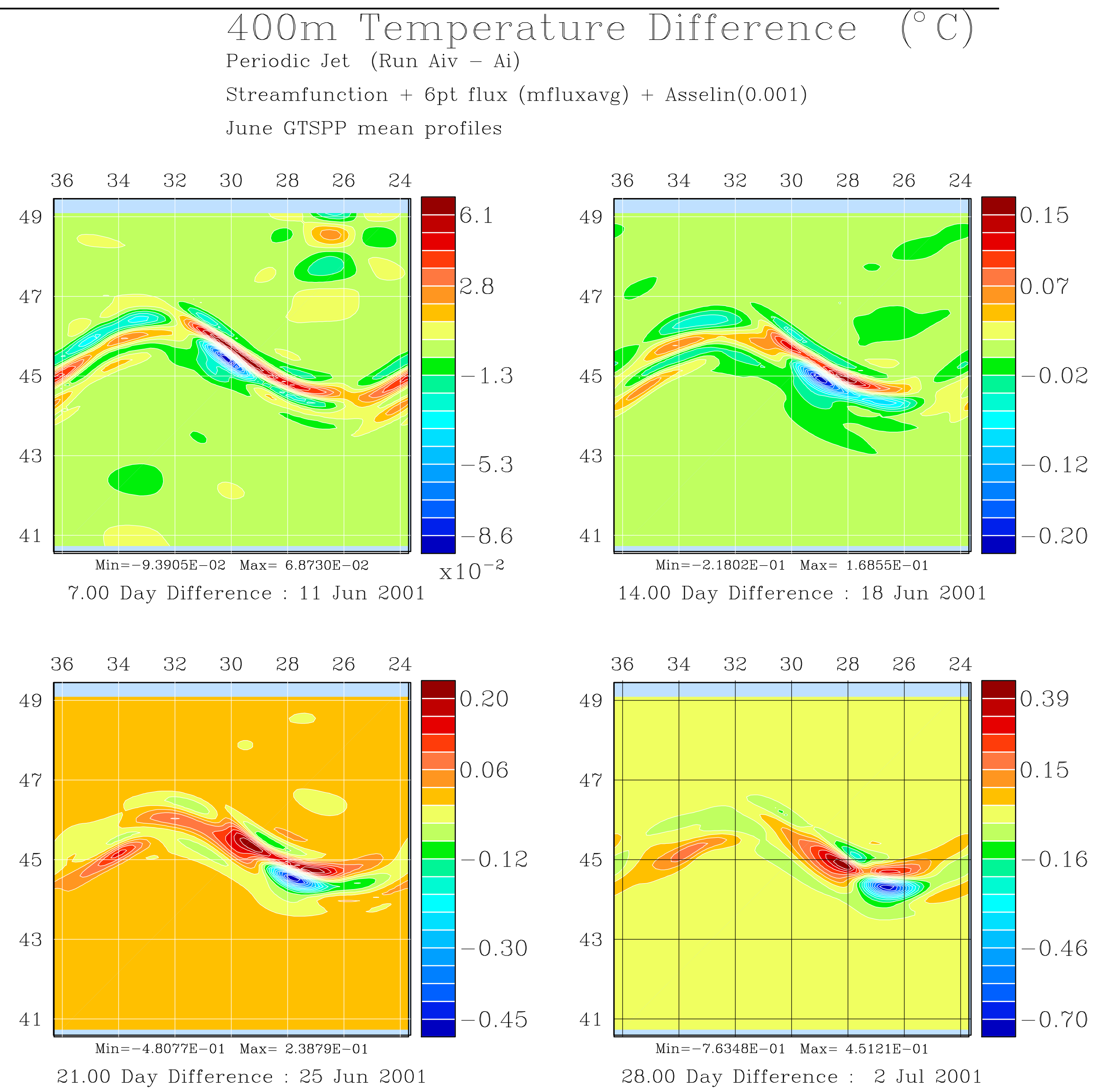


Figure 20 : Mid–level comparison of baseline simulation to simulation with divergence operator from surface pressure formulation plus Asselin filter (CPP options `mfluxavg` + `AsselinFilt`).

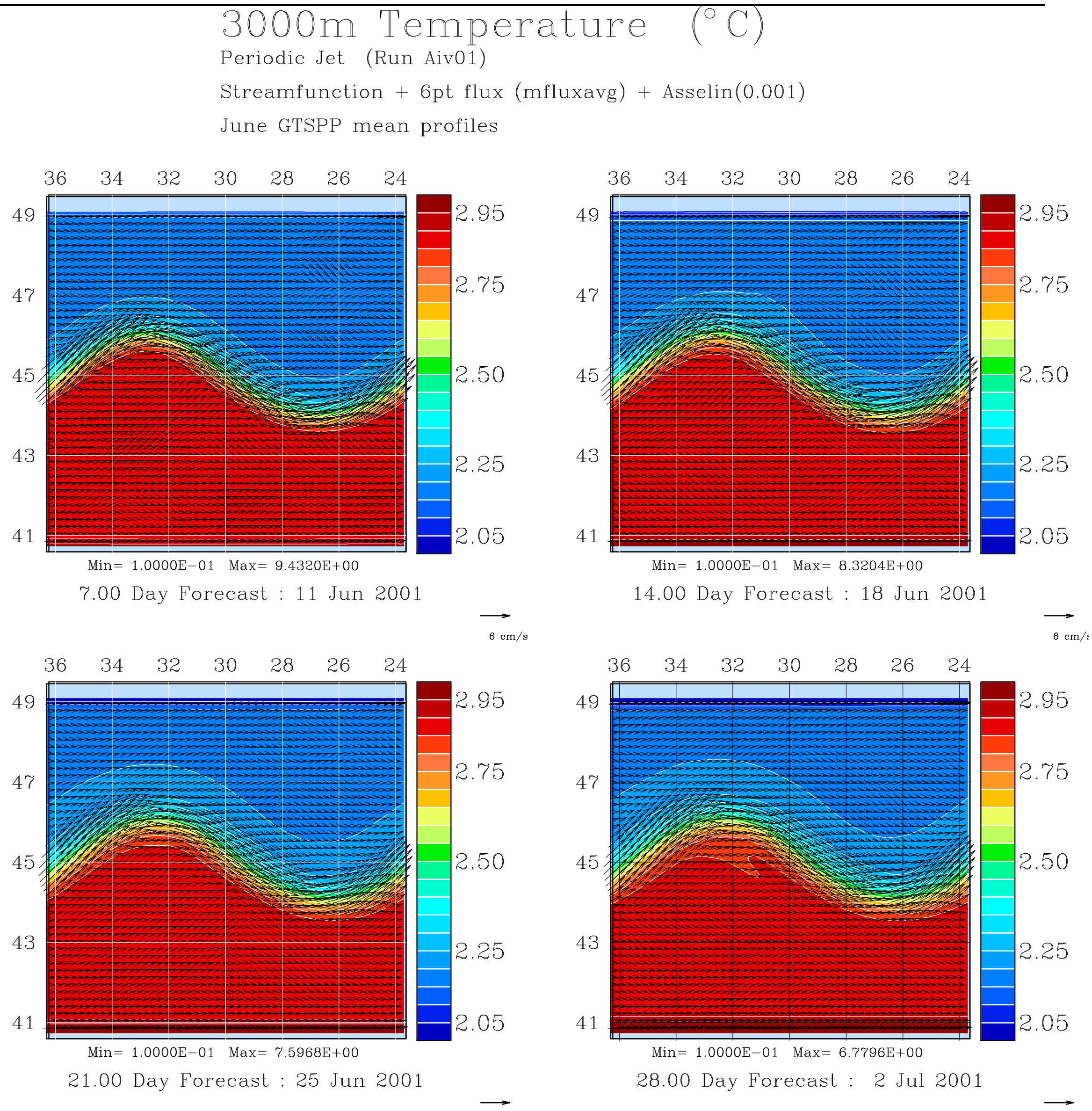


Figure 21 : Streamfunction formulation with divergence operator from surface pressure formulation plus Asselin filter. Deep temperature with total velocity vectors.

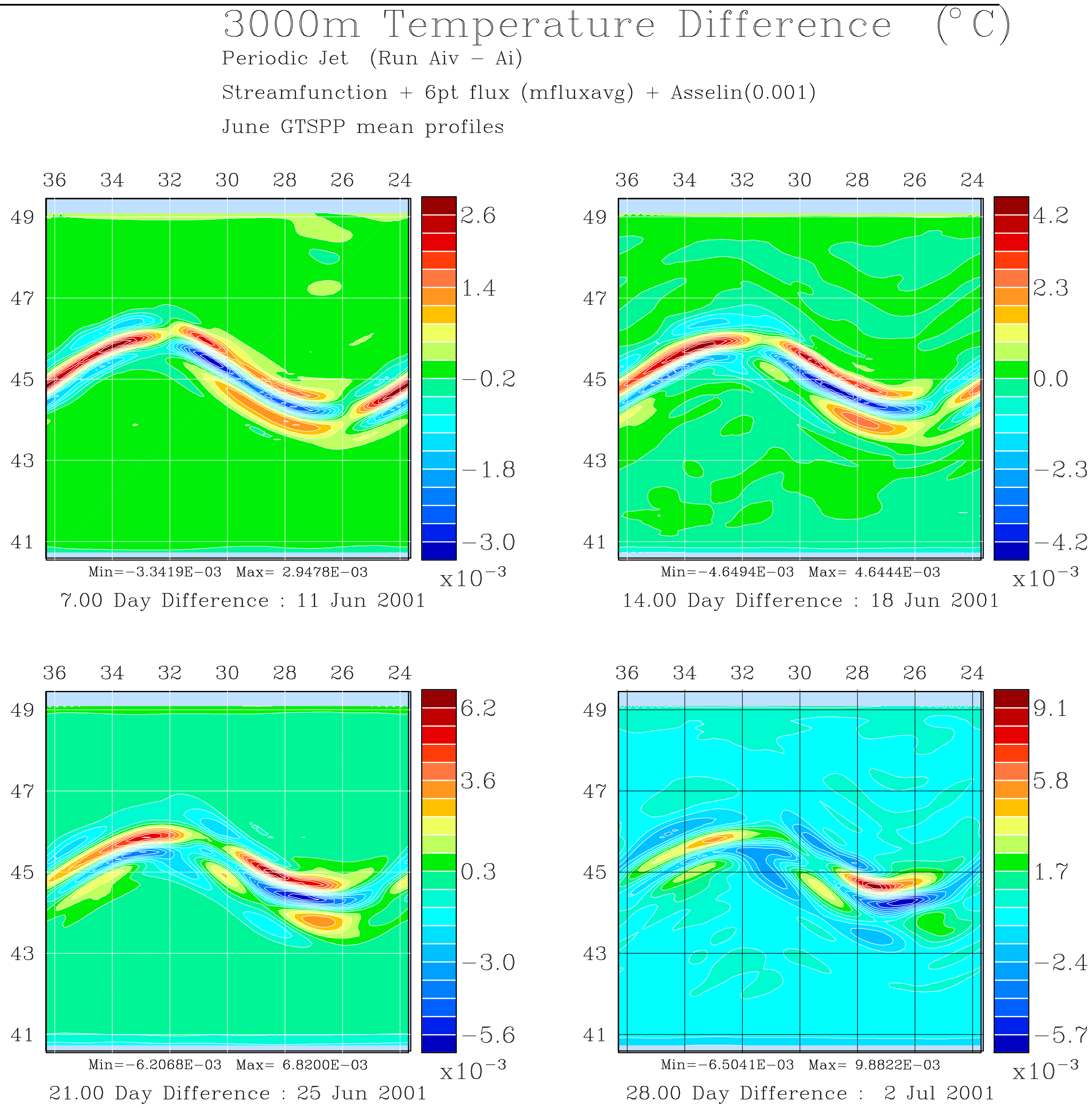


Figure 22 : Deep comparison of baseline simulation to simulation with divergence operator from surface pressure formulation plus Asselin filter (CPP options `mfluxavg` + `Asselin-Filt`).

(v) add Pre–conditioned Conjugate Gradient solver. Compare both solutions and timings.

To more easily and efficiently implement the new Poisson problem for the surface pressure formulation, Lozano & Lanerolle have introduced a sparse matrix, preconditioned conjugate gradient solver (SparsKit).

A comparison between *(i)* and *(v)* is presented in figures 23–30. The pre–conditioned conjugate gradient solver does not appear to affect the solution. The results are virtually identical to those of *(iv)*.

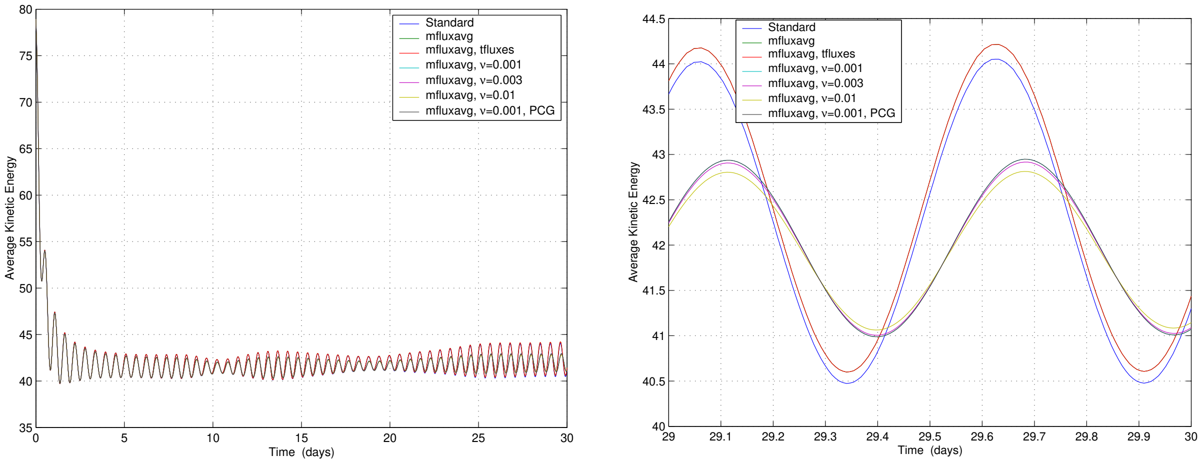


Figure 23 : Kinetic Energy comparison between baseline simulation *(i)* versus 6 point flux algorithm, Asselin filter & pre–conditioned conjugate gradient solver *(v)* (CPP options `mfluxavg`, `AsselinFilt` + `pcg5`).

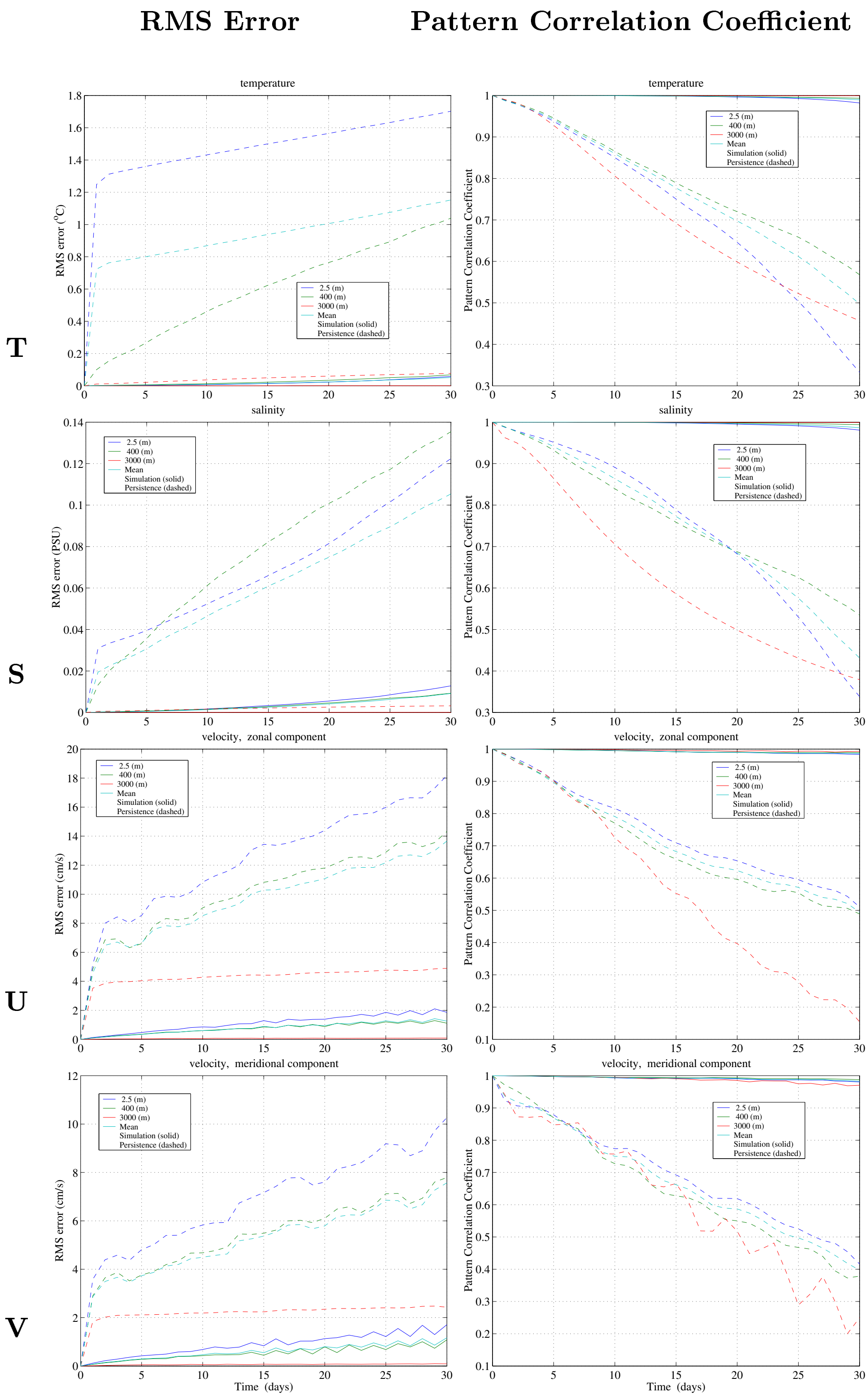


Figure 24: Comparison of baseline run *(i)* versus 6 point flux algorithm, Asselin Filter and preconditioned conjugate gradient solver. *(v)*

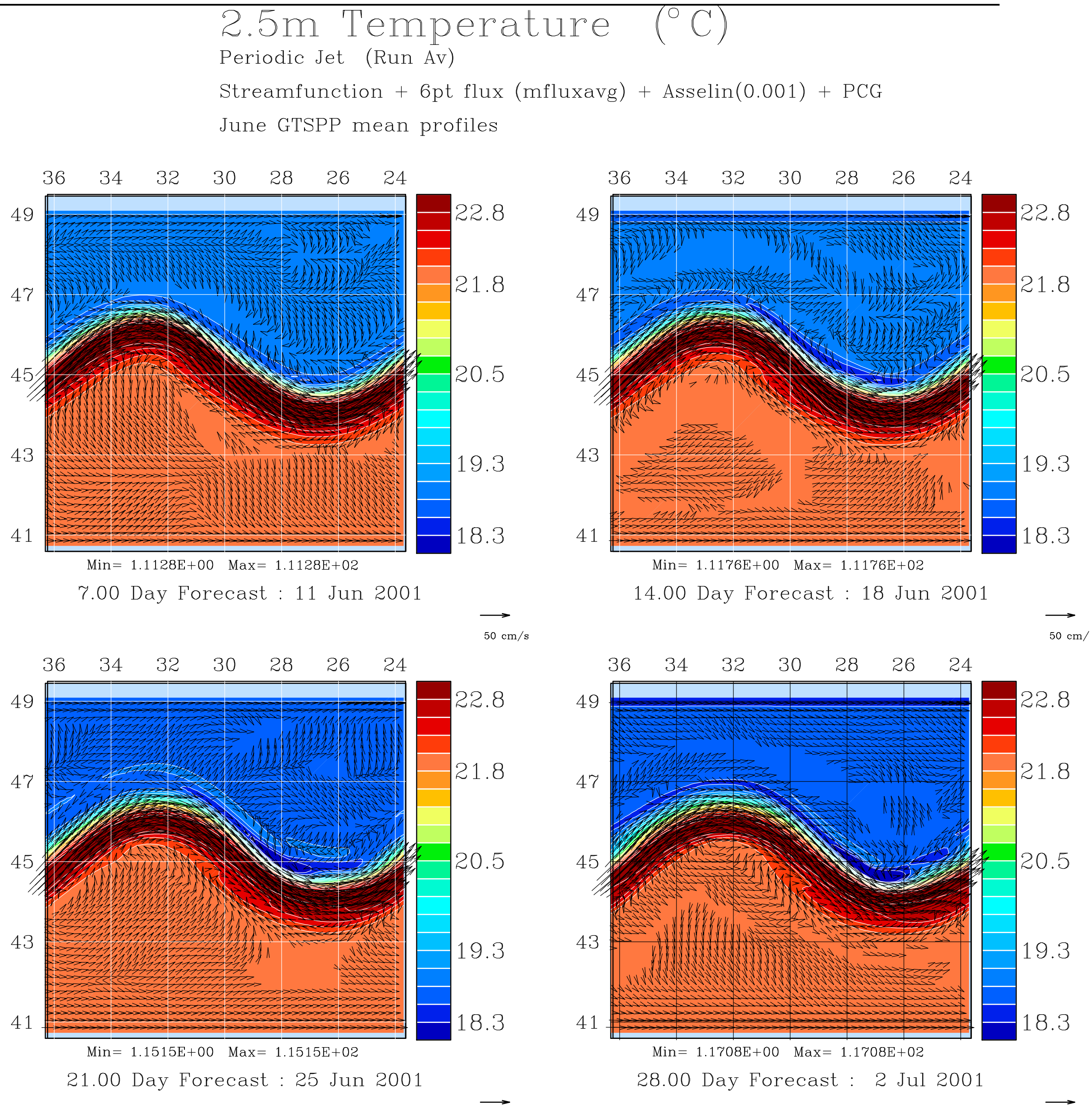


Figure 25 : Streamfunction formulation with divergence operator from surface pressure formulation, Asselin filter and preconditioned conjugate gradient solver. Near surface temperature with total velocity vectors.

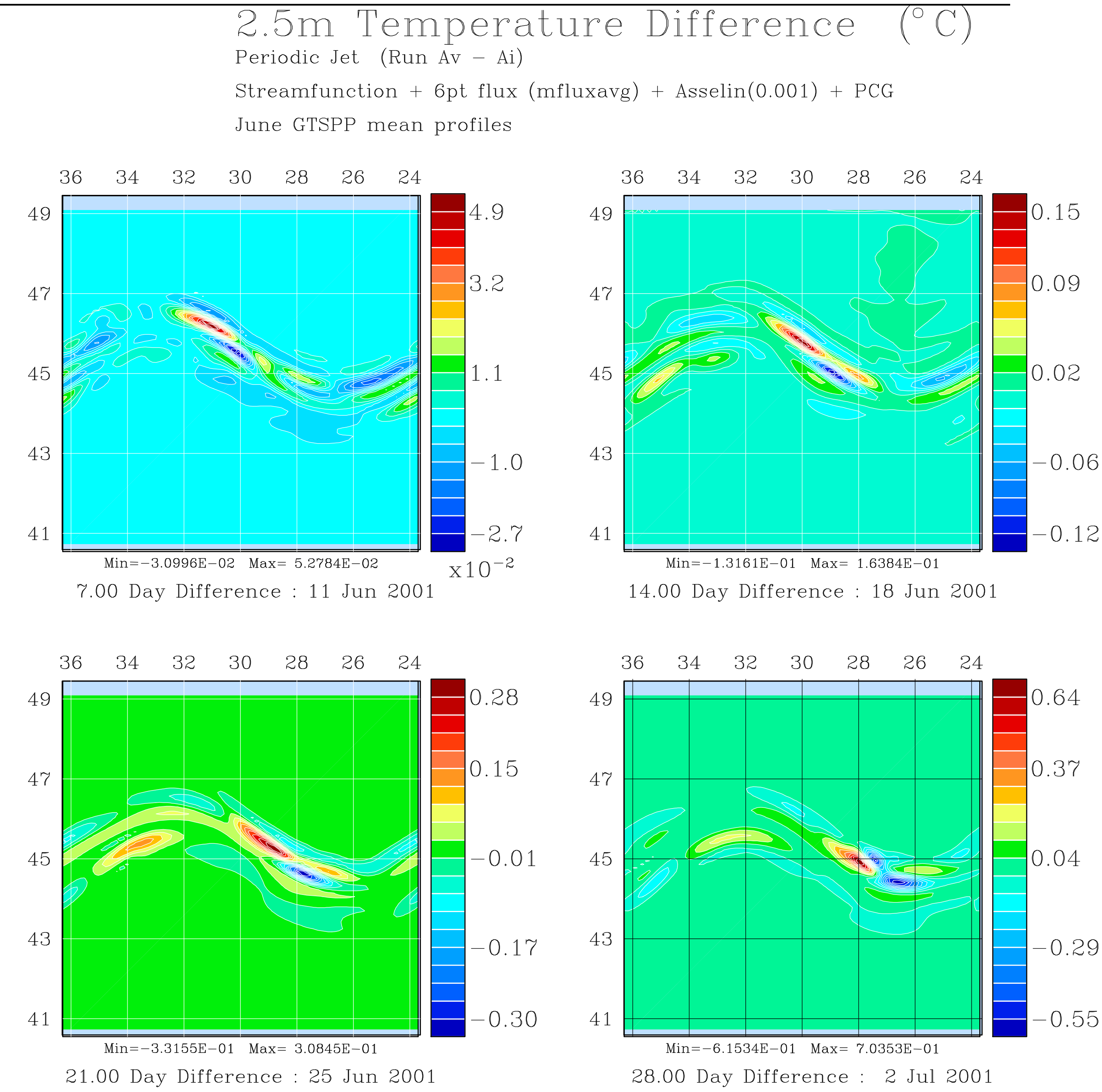


Figure 26 : Near Surface comparison of baseline simulation to simulation with divergence operator from surface pressure formulation plus Asselin filter and preconditioned conjugate gradient solver (CPP options `mfluxavg`, `AsselinFilt` + `pcg5`).

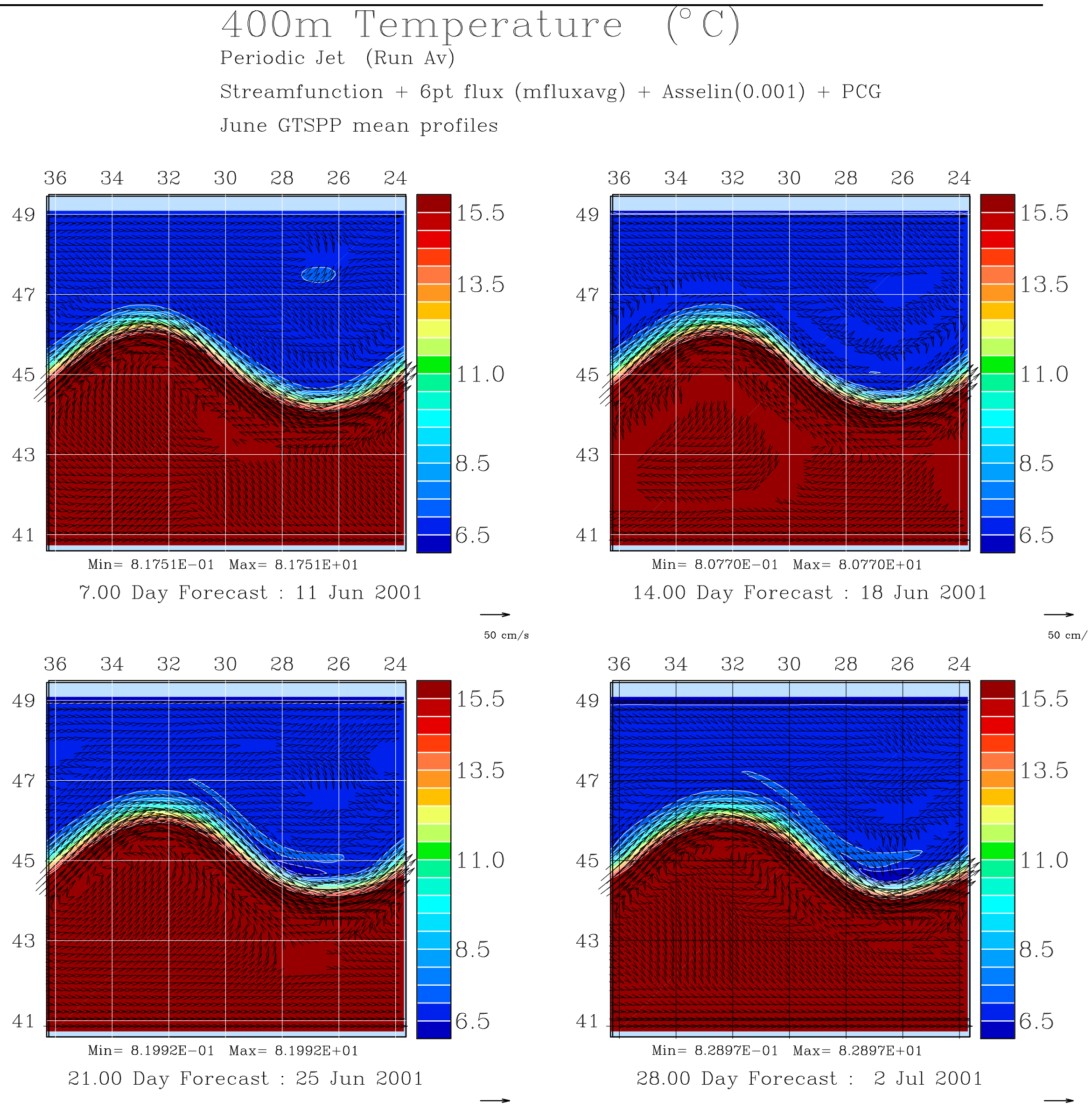


Figure 27 : Streamfunction formulation with divergence operator from surface pressure formulation, Asselin filter and preconditioned conjugate gradient solver. Mid–level temperature with total velocity vectors.

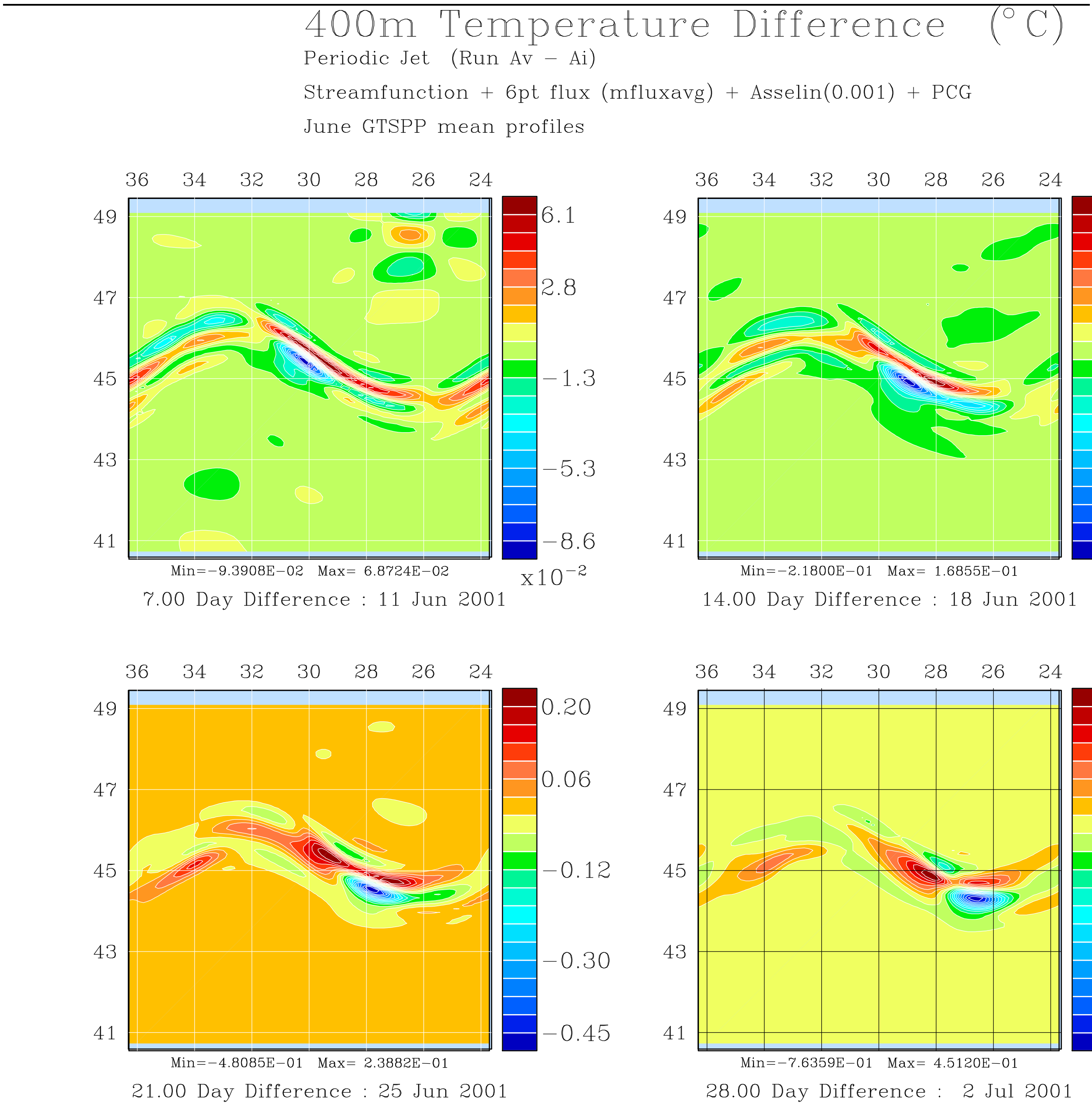


Figure 28 : Mid–level comparison of baseline simulation to simulation with divergence operator from surface pressure formulation plus Asselin filter and preconditioned conjugate gradient solver (CPP options `mfluxavg`, `AsselinFilt` + `pcg5`).

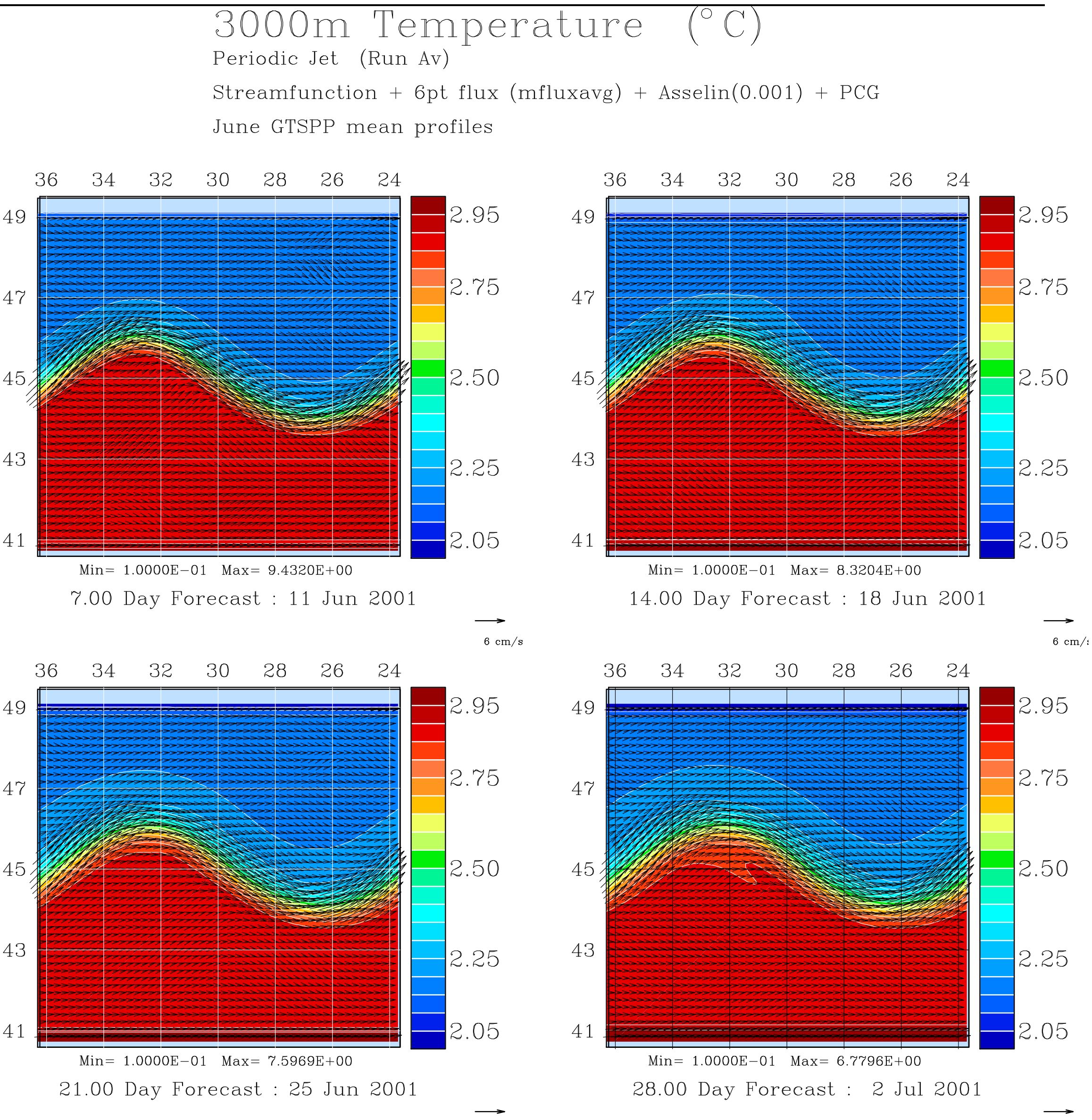


Figure 29 : Streamfunction formulation with divergence operator from surface pressure formulation, Asselin filter and preconditioned conjugate gradient solver. Deep temperature with total velocity vectors.

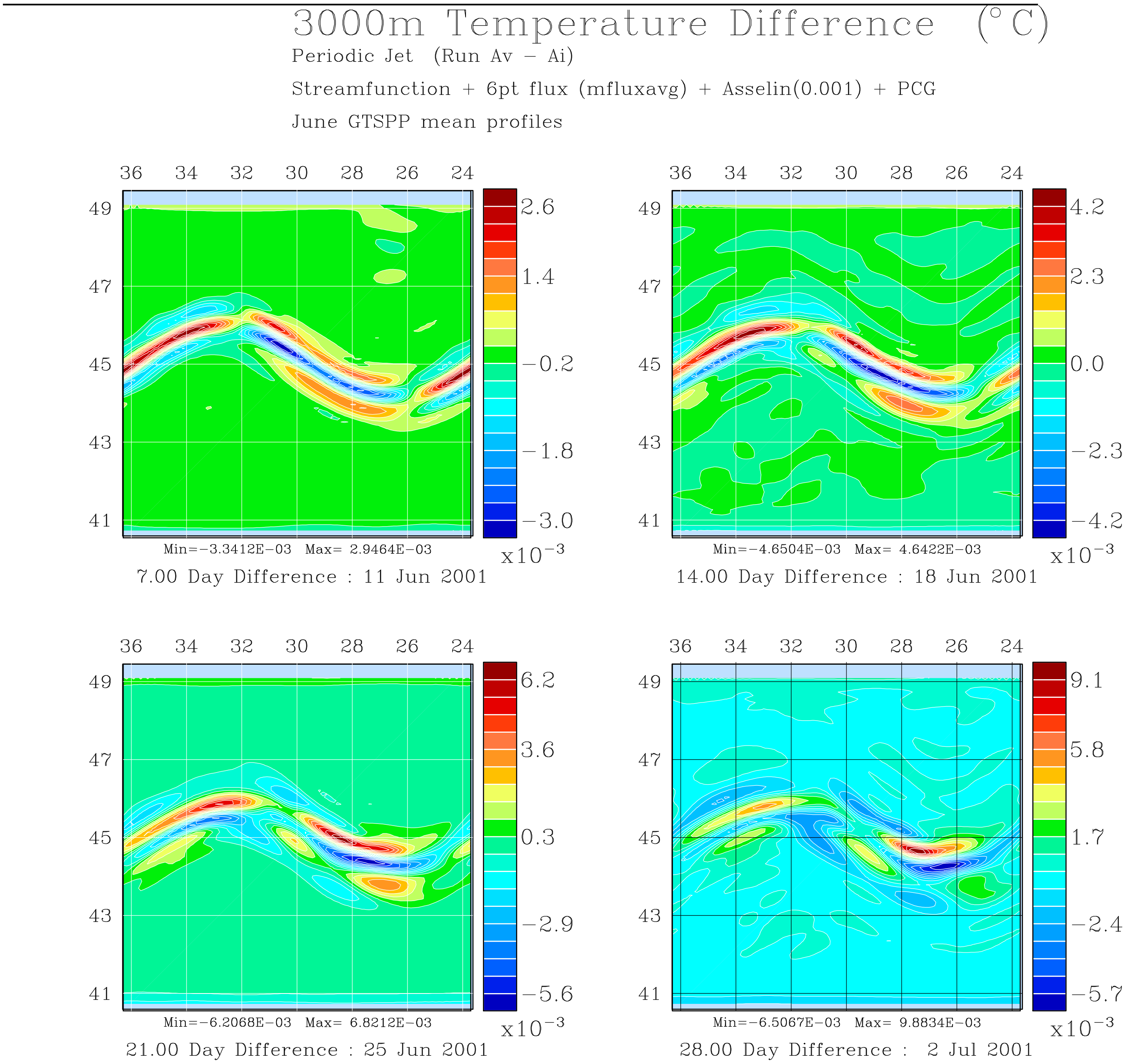


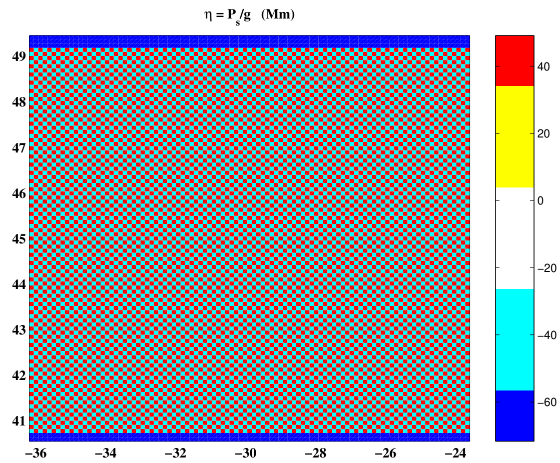
Figure 30 : Deep comparison of baseline simulation to simulation with divergence operator from surface pressure formulation plus Asselin filter and preconditioned conjugate gradient solver (CPP options `mfluxavg`, `AsselinFilt` + `pcg5`).

(B) Surface Pressure formulation – Rigid Lid

Initialization Procedure

The initialization procedure is described in the P^2S^3 write–up. The various options (initial guess & Dirichlet boundary conditions) do not really matter to the periodic channel domain. For these experiments, the surface streamfunction initialization was arbitrarily chosen.

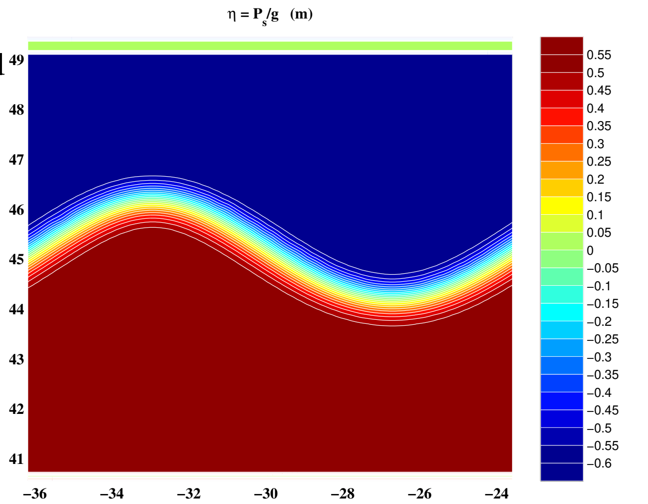
Tuning Preconditioning Parameters



The first issue to arise in setting up the surface pressure initialization is the existence of a computational mode. The discretized Laplacian operator for the surface pressure formulation primarily relies on information from a "times" stencil

$$\begin{array}{ccc} P_{i-1,j+1} & & P_{i+1,j+1} \\ & P_{i,j} & \\ P_{i-1,j-1} & & P_{i+1,j-1} \end{array}$$

This discrete operator has a computational mode in a "checker board" pattern. For most applications this is merely another source of noise. For this periodic channel the computational mode is dominant. The conjugate gradient solver fails to converge, even to produce the initial field. The results after 6000 iterations are presented in the above figure, in which (for more intuitive horror) the kinematic surface pressure is rescaled to surface elevation in megameters (megameters because the color bar kept losing it's scaling factor). The original signal is lost, so naive filtering is out. Fortunately, Lozano included an option of introducing a "Laplacian filter" directly into the operator. This is accomplished by adding a term like



$$\lambda \begin{pmatrix} -\frac{1}{2}P_{i-1,j+1} & P_{i,j+1} & -\frac{1}{2}P_{i+1,j+1} \\ P_{i-1,j} & -2P_{i,j} & P_{i+1,j} \\ -\frac{1}{2}P_{i-1,j-1} & P_{i,j-1} & -\frac{1}{2}P_{i+1,j-1} \end{pmatrix}$$

to each row, where $\lambda = 10^{-6}$. For comparison, the original matrix has coefficients $O\left(10^{-1}\right)$. This is enough to allow a reasonable solution in the *unpreconditioned* conjugate gradient algorithm (right). The matrix is perturbed enough so that the preconditioning algorithm fails, hence the surface pressure results in this test don't use it. Note that the surface elevation signal, ± 50 cm, is on the same order as the estimated geostrophic signal in the Gulf Stream, 30 cm (Calman 1985).

(i) Surface Streamfunction Initialization (`psurf` = 0).

By far the largest differences in this series occur in the comparison of the streamfunction to the surface pressure formulations. The kinetic energy (figure 33) shows the same oscillatory behavior, but with a gradual decay absent from the streamfunction cases.

The skill metrics (figure 34) still show a generally favorable comparison to persistence, but with far and away the largest RMS and smallest PCC in this series.

The temperature and difference plots (figures 35–40) indicate that part of the poor showing in the skill metrics is due to a phase error. The difference fields indicate a relative westward shift of the front, which can be seen in the temperature field plots.

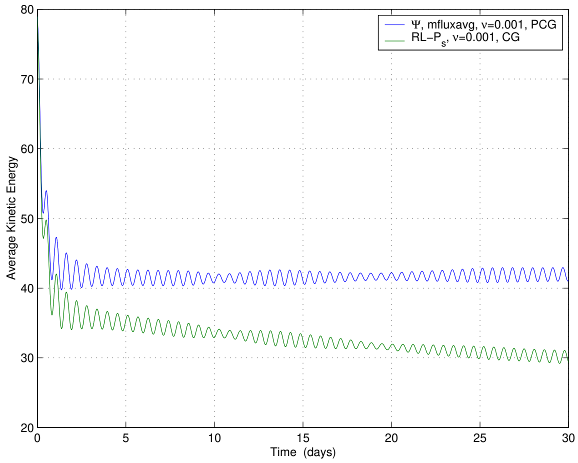


Figure 33 : Kinetic Energy comparison between "most similar" rigid–lid streamfunction simulation *(Av)* to rigid–lid surface pressure simulation *(Bi)* with a constant extrapolation boundary condition on vertically integrated forcing..

RMS Error　　　　Pattern Correlation Coefficient

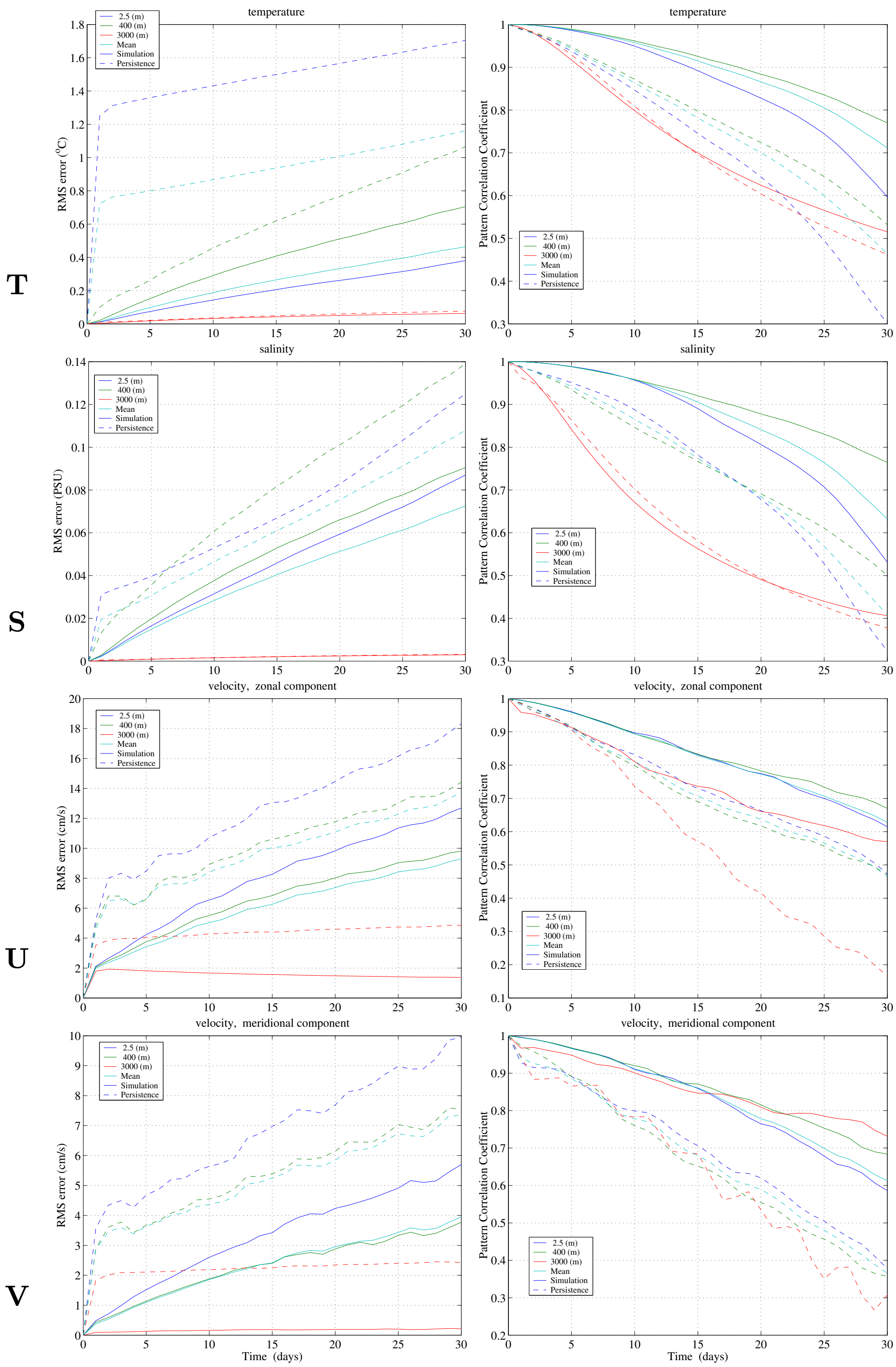


Figure 34: Comparison of "most similar" rigid–lid streamfunction simulation *(Av)* to rigid–lid surface pressure simulation *(Bi)* with a constant extrapolation boundary condition on vertically integrated forcing.

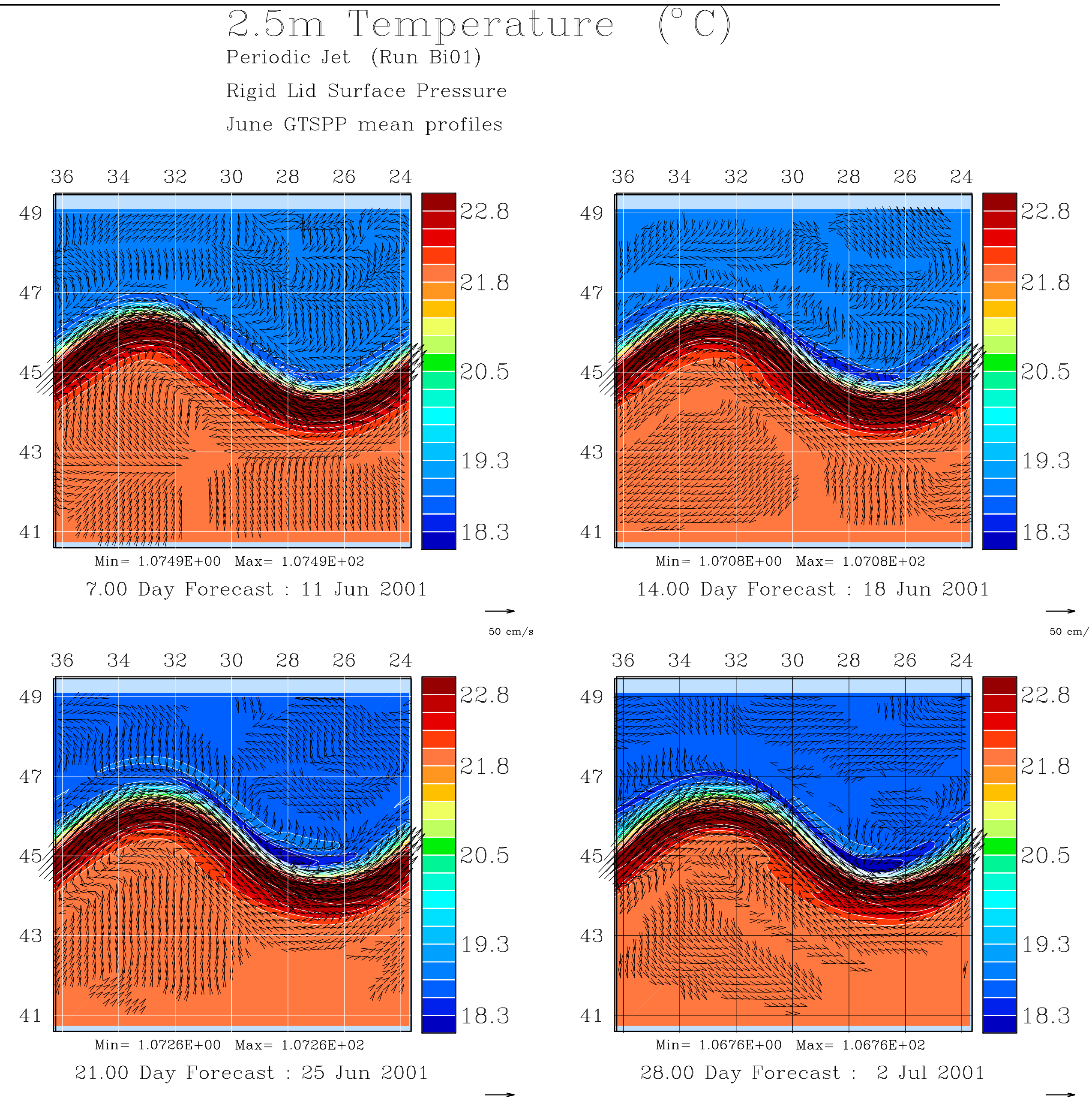


Figure 35 : Rigid–lid, surface pressure formulation from surface streamfunction initialization (`psurf` = 0). Near surface temperature with total velocity vectors.

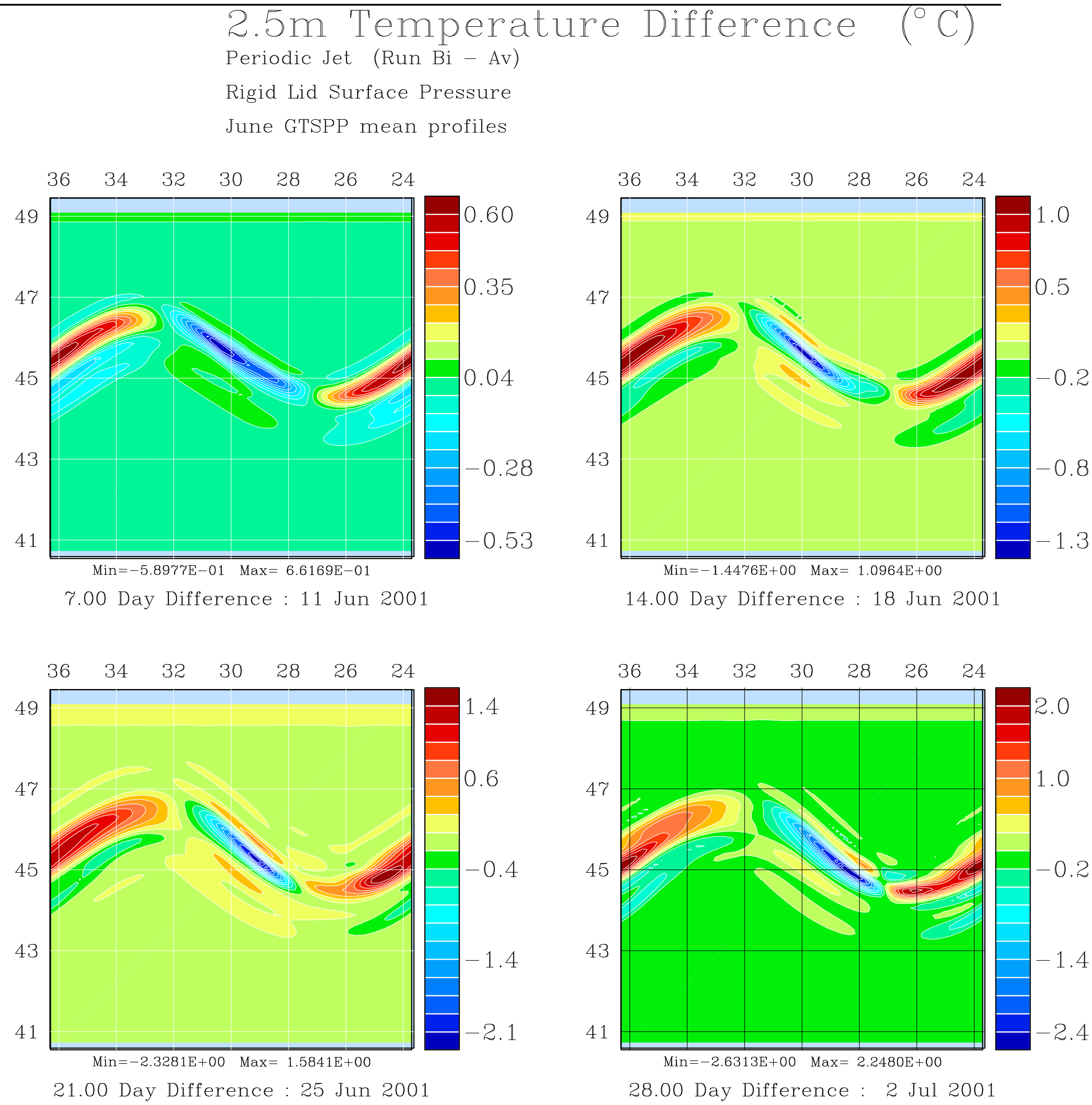


Figure 36 : Near Surface comparison of streamfunction formulation with divergence operator from surface pressure formulation plus Asselin filter (CPP options `mfluxavg` + `AsselinFilt`) to rigid–lid, surface pressure formulation from surface streamfunction initialization (`psurf` = 0).

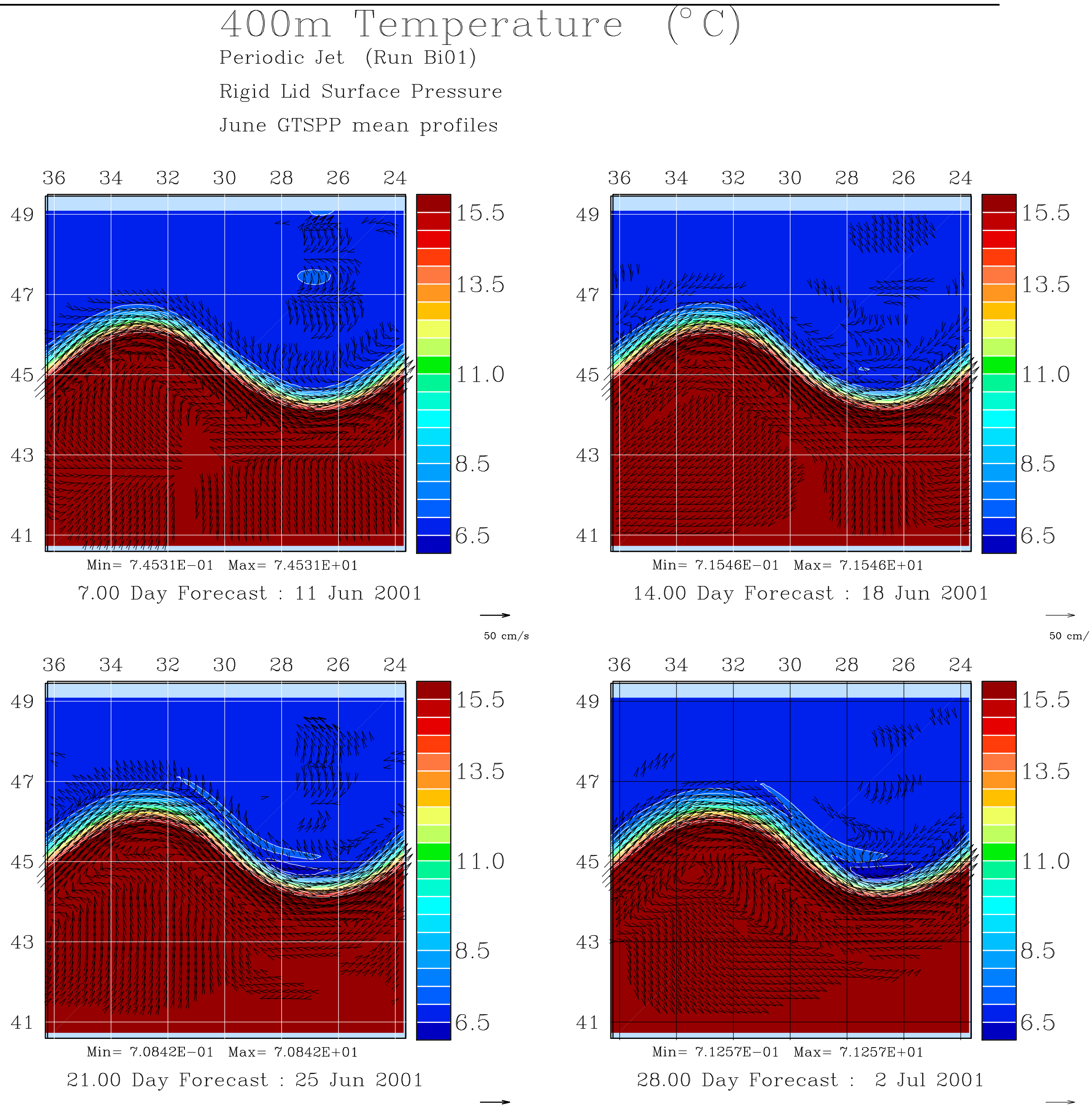


Figure 37 : Rigid–lid, surface pressure formulation from surface streamfunction initialization (`psurf` = 0). Mid–level temperature with total velocity vectors.

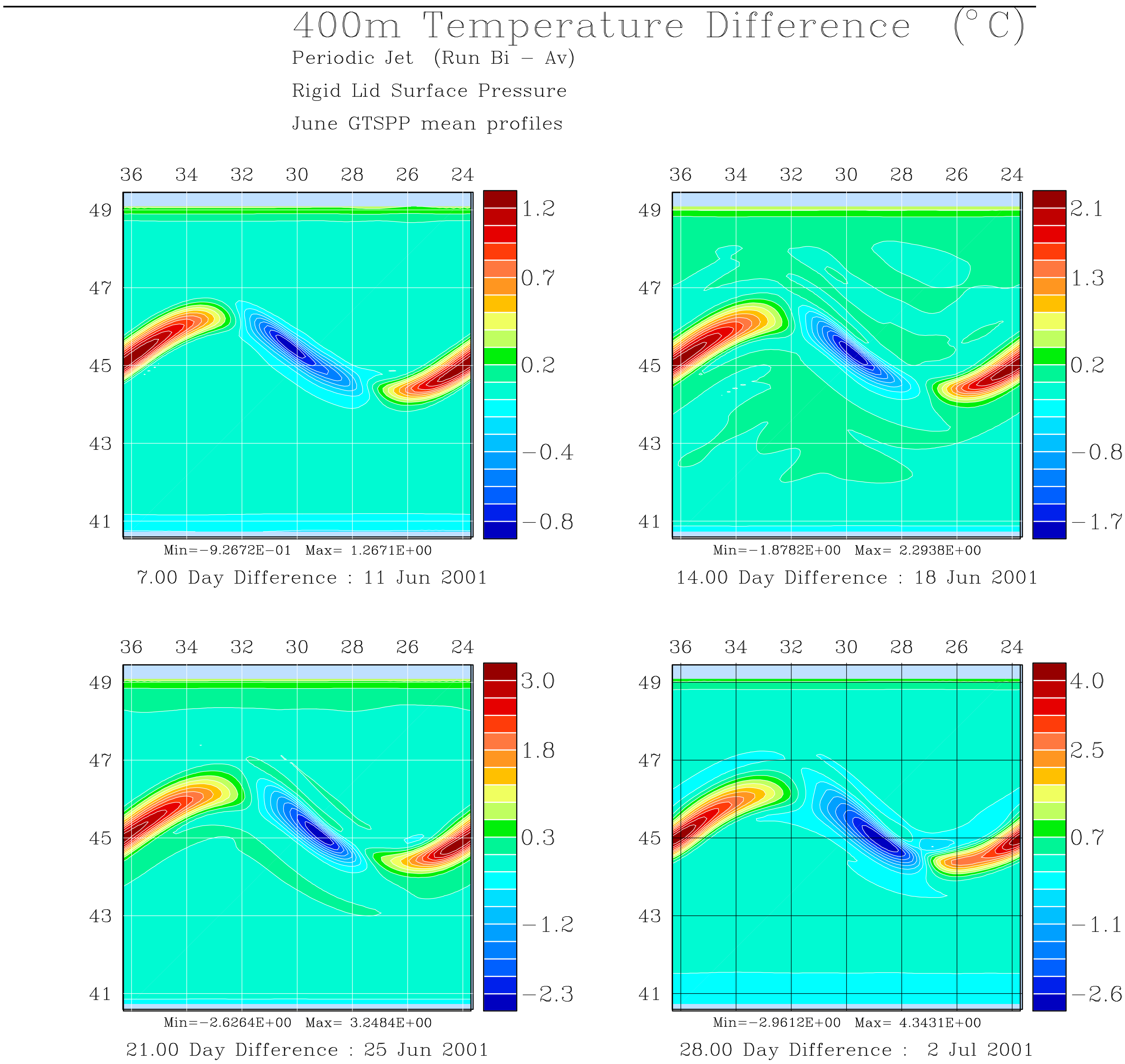


Figure 38 : Mid–level comparison of streamfunction formulation with divergence operator from surface pressure formulation plus Asselin filter (CPP options `mfluxavg` + `Asselin-Filt`) to rigid–lid, surface pressure formulation from surface streamfunction initialization (`psurf` = 0).

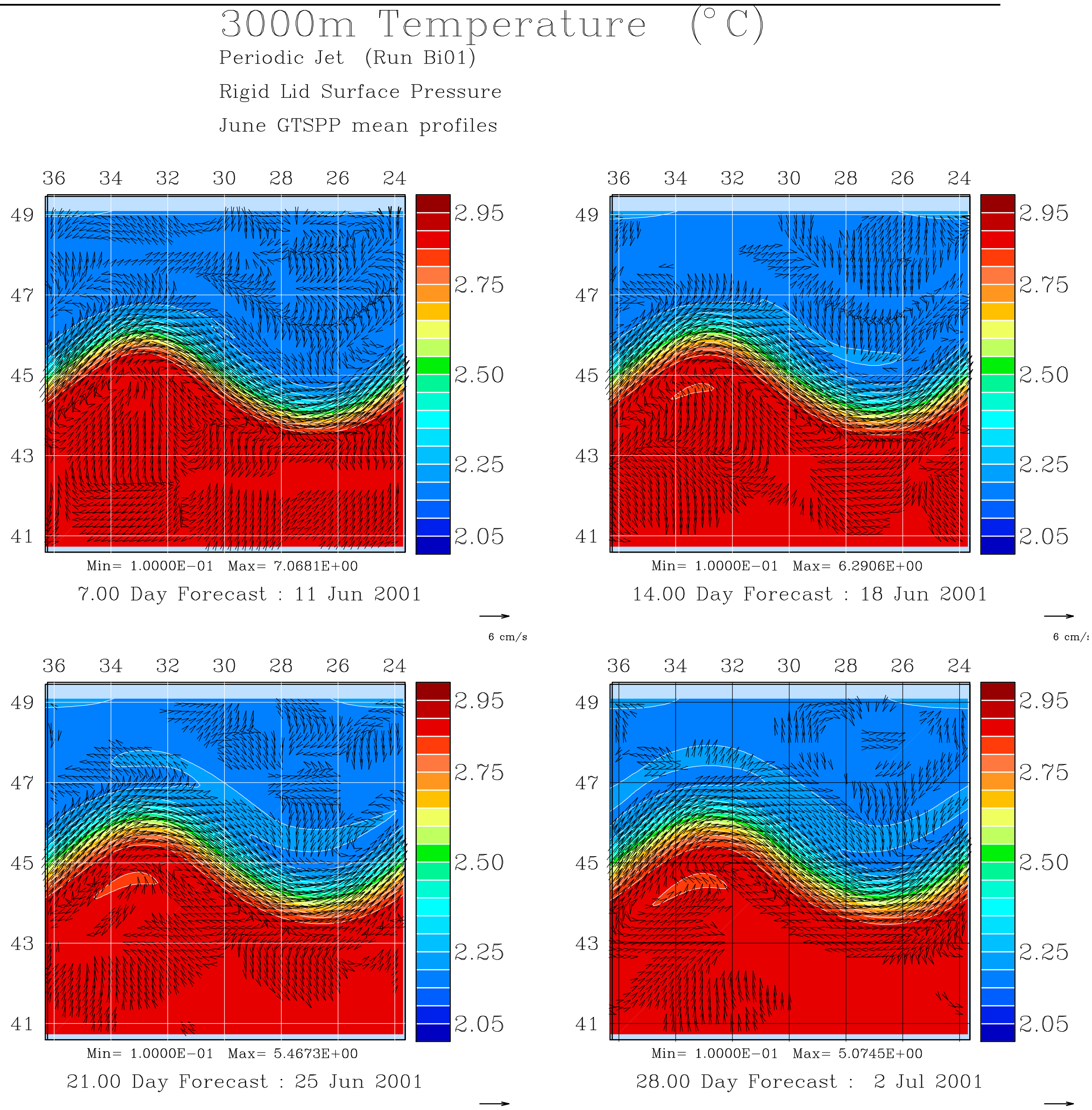


Figure 39 : Rigid–lid, surface pressure formulation from surface streamfunction initialization (`psurf` = 0). Deep temperature with total velocity vectors.

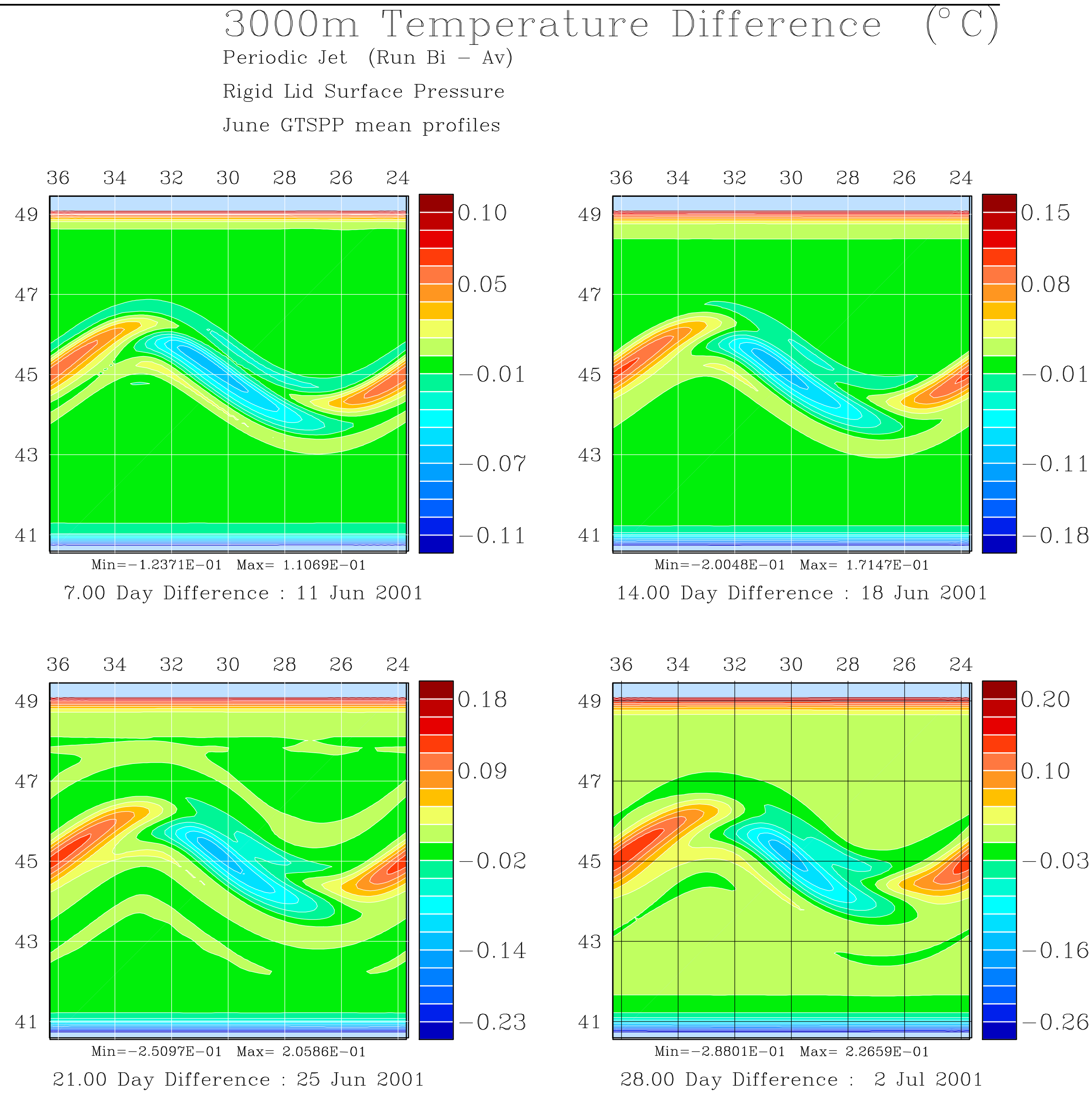


Figure 40 : Deep comparison of streamfunction formulation with divergence operator from surface pressure formulation plus Asselin filter (CPP options `mfluxavg` + `AsselinFilt`) to rigid–lid, surface pressure formulation from surface streamfunction initialization (`psurf` = 0).

(C) Surface Pressure formulation – Free Surface *(i–iv)* as in *(B)*.

As in the rigid–lid case, only one initialization was needed.

One note: at press time, the plotting and skill metric codes were ignoring the effect of the free surface on the model levels. For this test, the free surface variability is in the range ± 50 cm and so is not a dominant effect.

(i) Surface Streamfunction Initialization (`psurf` = 0).

The free surface results and comparisons to the rigid lid are presented in figures 41–48. The kinetic energies, the RMS & PCC statistics and the magnitudes of the difference fields all point to a greater similarity between the two surface pressure formulations that between either and the streamfunction formulation.

The kinetic energy comparison in figure 41 shows the emergence of an odd–even decoupling between the time steps. This would indicate that the free surface case could probably use a larger value for the Asselin filter parameter. Visually filtering the free surface results shows the free surface KE leads the rigid-lid KE by ∼70 minutes and is depressed by a value of $\sim \frac{1}{2}$.

The main patterns in the difference fields indicate another westward shift of the free surface results relative to the rigid–lid results. The 2m difference fields also indicate a secondary binary difference depending on whether a point is north or south of the front. This is probably due to the plotting codes current inability to recognize and deal with the altered vertical levels in the free surface results.

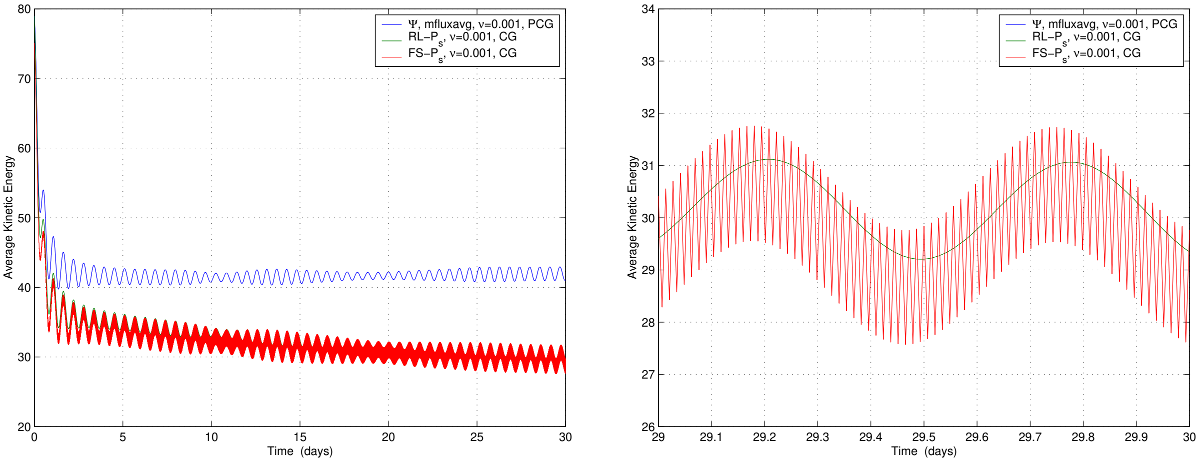


Figure 41 : Kinetic Energy comparison between simulations *(Av)*, *(Bi)* and *(Ci)*.

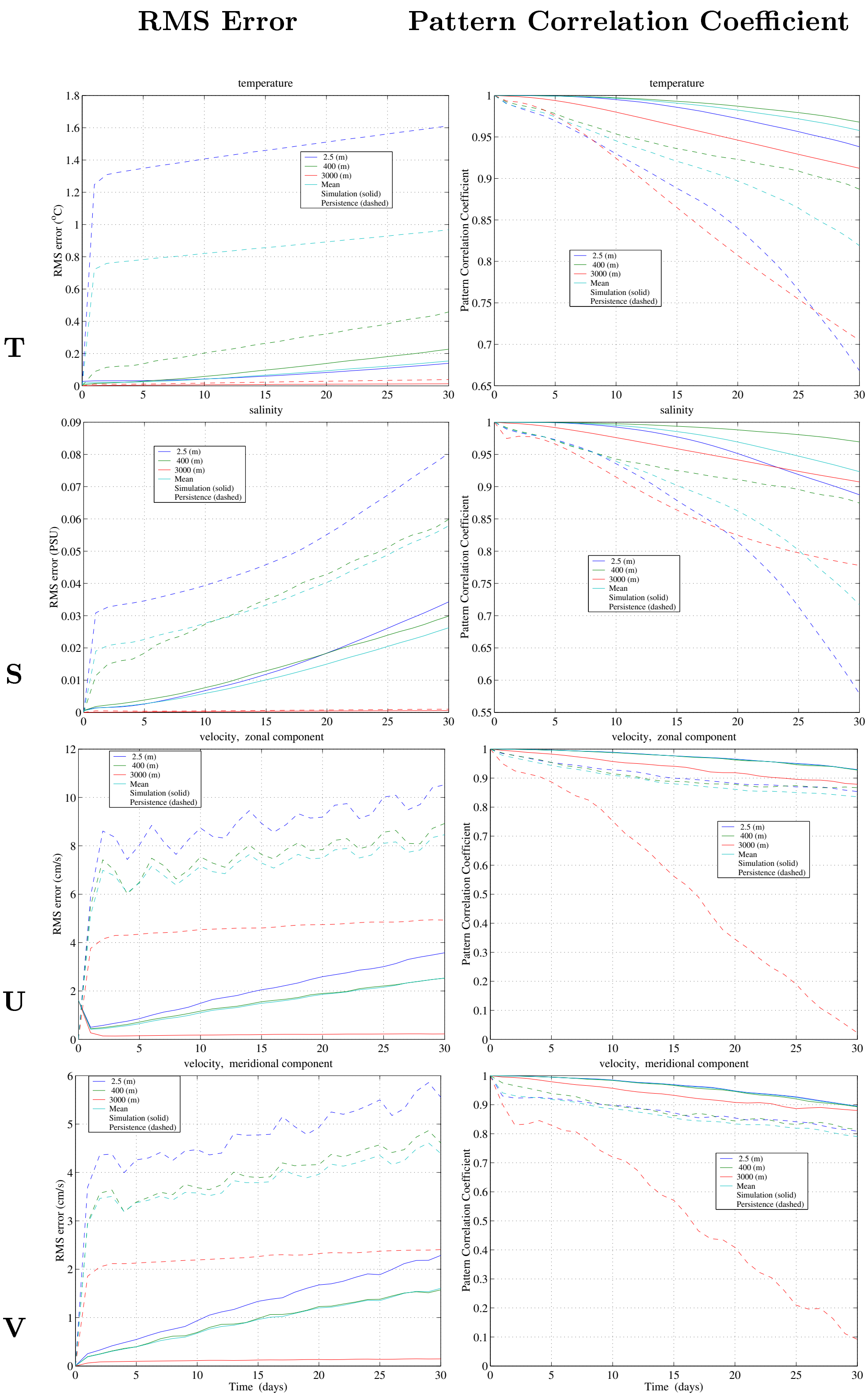


Figure 42: Comparison of rigid–lid surface pressure simulation *(Bi)* to free surface, surface pressure simulation *(Ci)*.

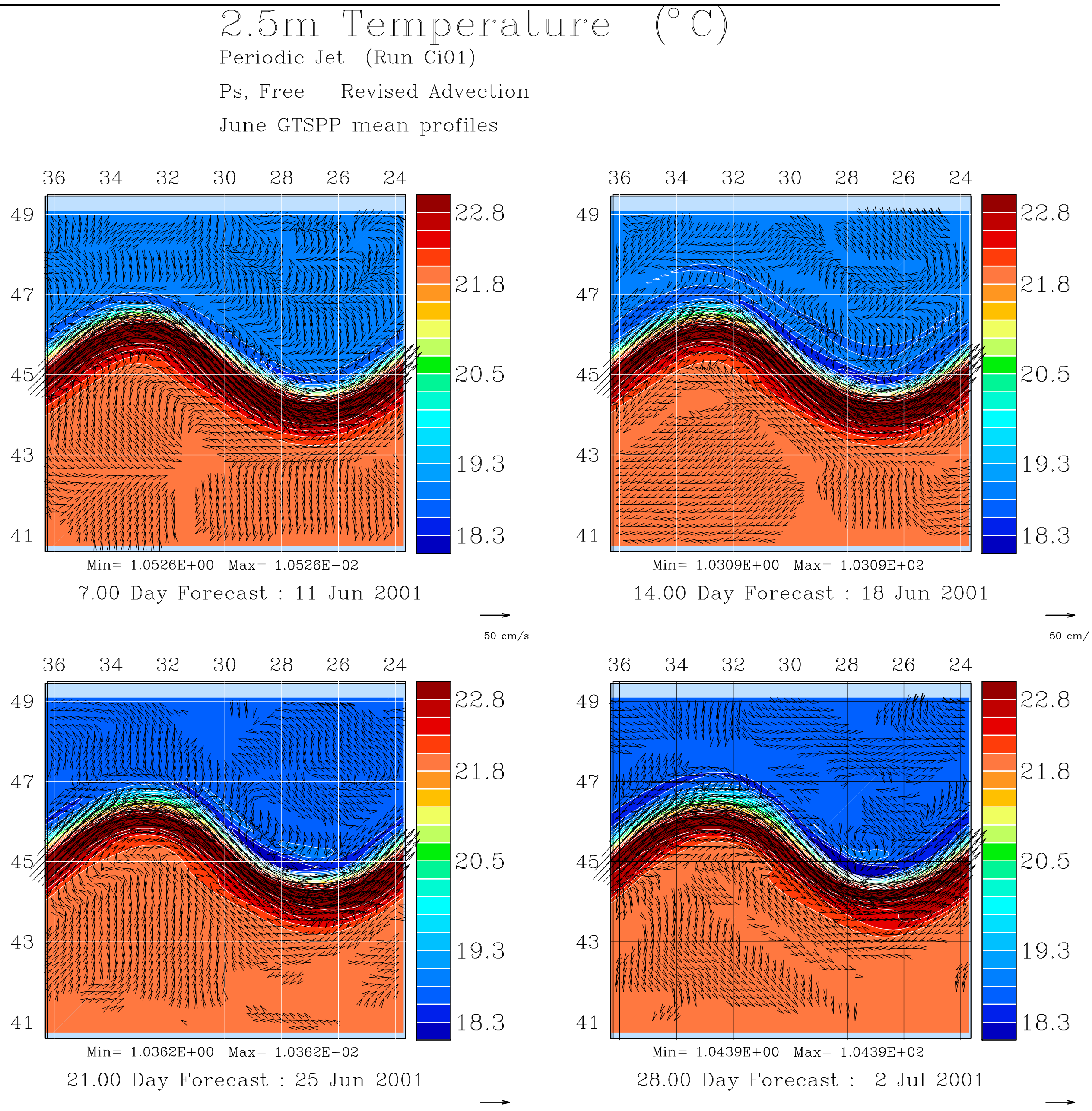


Figure 43 : Free surface, surface pressure formulation from surface streamfunction initialization (`psurf` = 0). Near surface temperature with total velocity vectors.

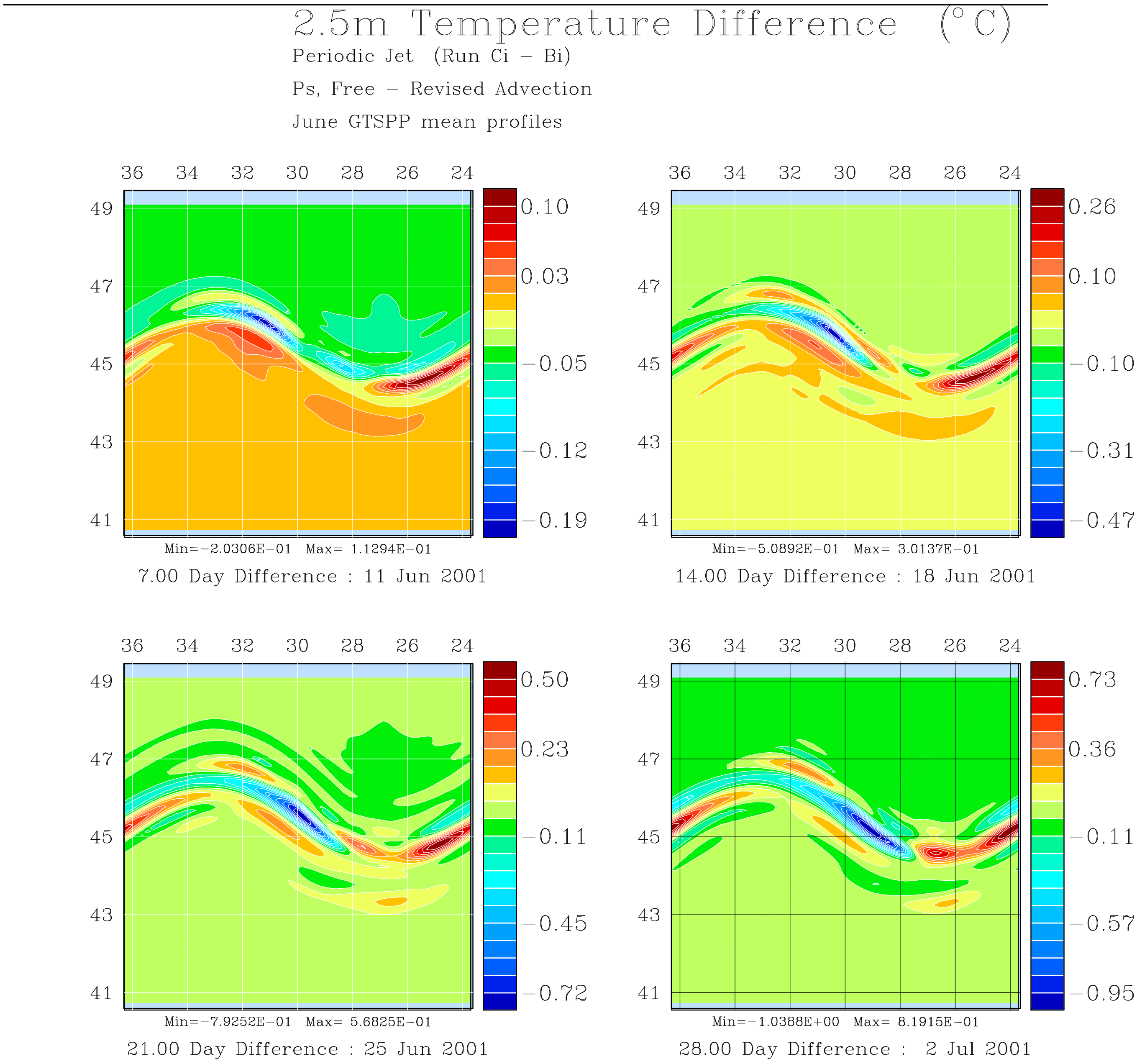


Figure 44 : Near Surface comparison of free surface, surface pressure formulation to rigid–lid, surface pressure formulation.

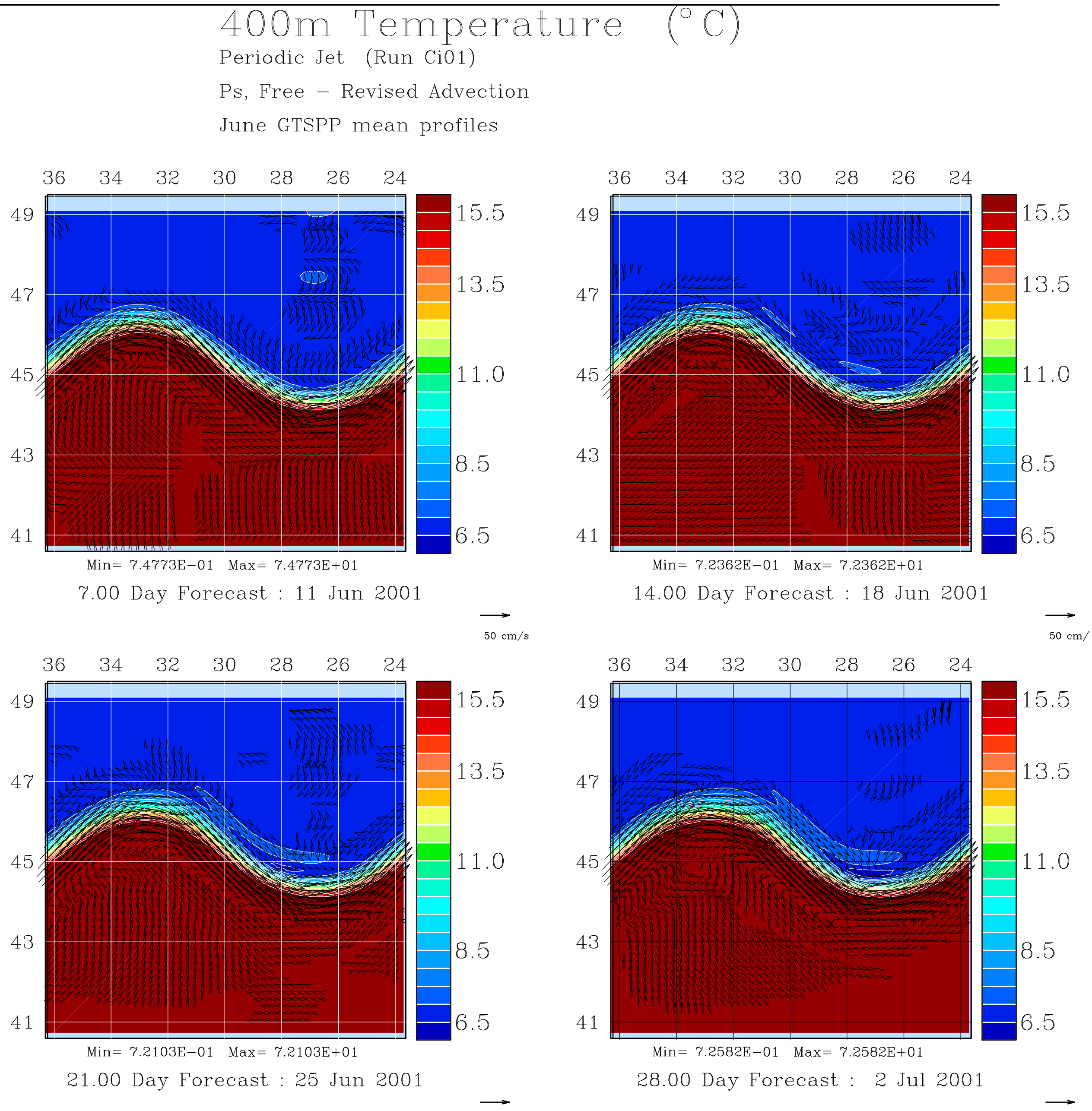


Figure 45 : Free surface, surface pressure formulation from surface streamfunction initialization (`psurf` = 0). Mid–level temperature with total velocity vectors.

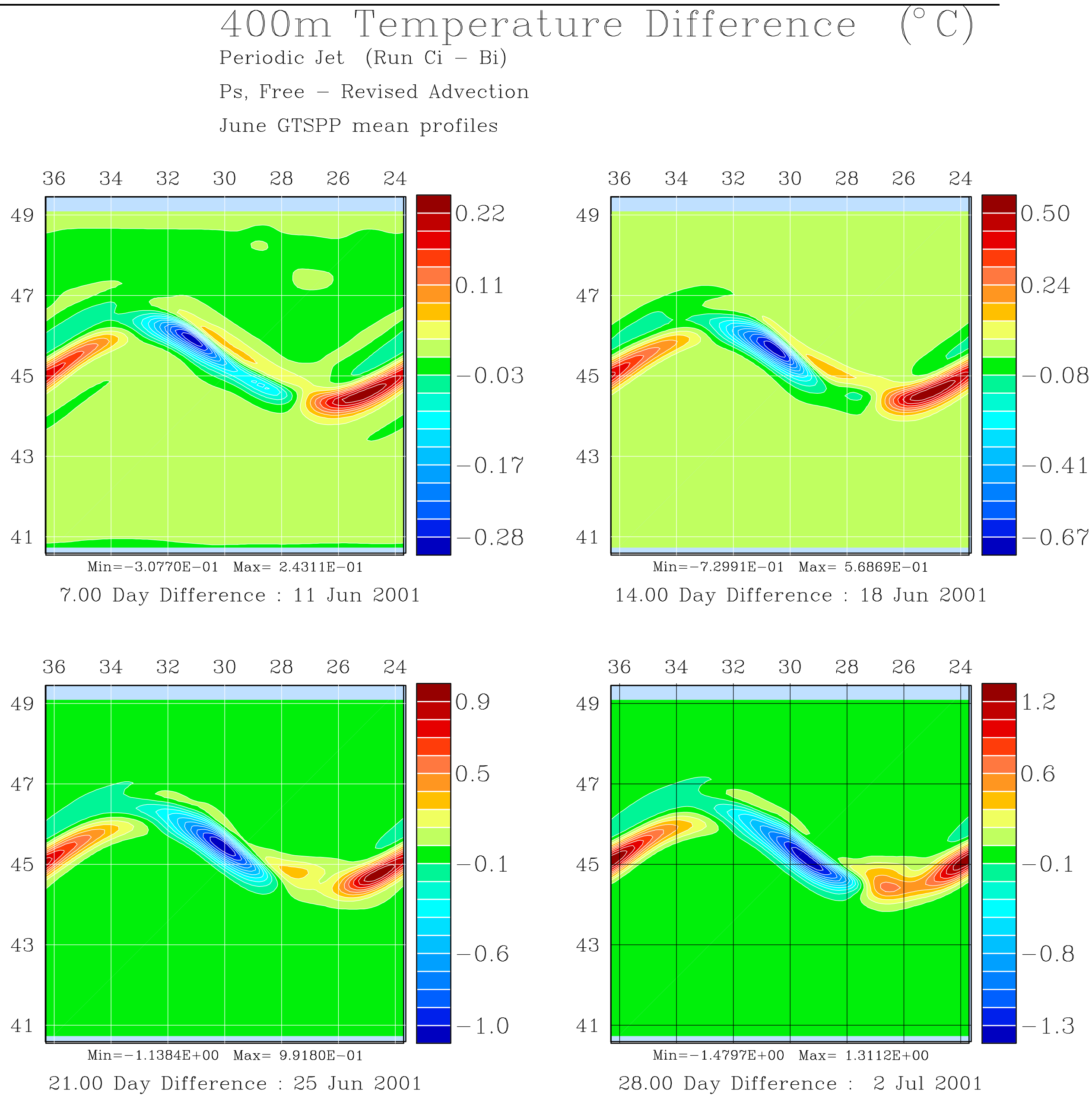


Figure 46 : Mid–level comparison of free surface, surface pressure formulation to rigid–lid, surface pressure formulation.

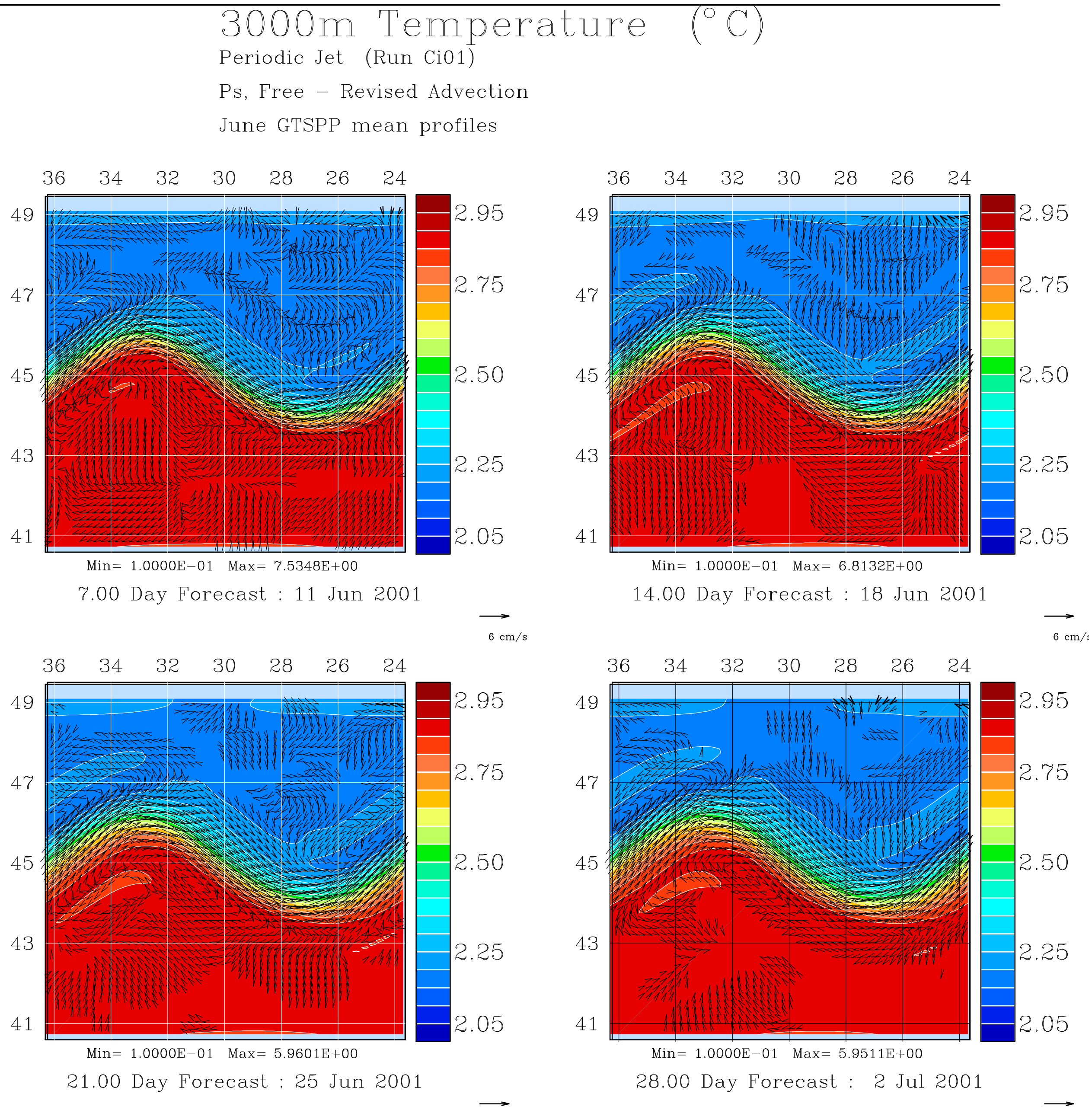


Figure 47 : Free surface, surface pressure formulation from surface streamfunction initialization (`psurf` = 0). Deep temperature with total velocity vectors.

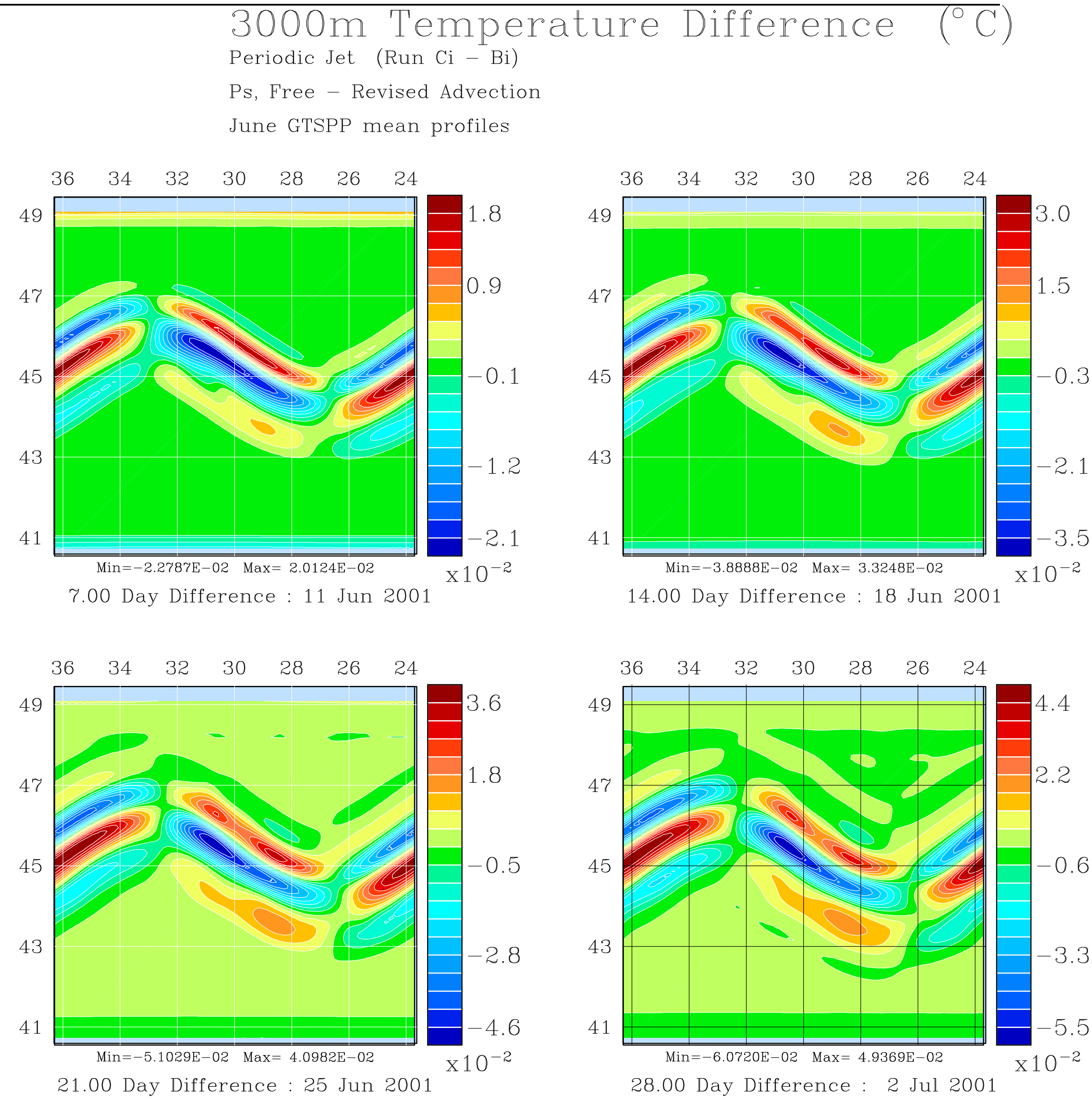


Figure 48 : Deep comparison of free surface, surface pressure formulation to rigid–lid, surface pressure formulation.

Summary

The GFD jet series demonstrates overall consistency between the transport streamfunction and surface pressure formulations. This idealized test in a domain without open boundaries provided a useful debugging tool for the code. The absence of open boundaries allowed the bulk of the code to be debugged sooner. Every non–boundary problem we corrected, we first solved in this series.

Notes

- The plotting and skill metric codes need to be cognizant of the changes to the vertical grid with the free surface.

References

Calman, J. (1985) Sea–Surface Height by GEOSAT–A Altimetry. In *Gulf Stream Workshop Proceedings*, D.R. Watts, ed. pp 143–168.