The upper ocean response to diurnal forcing in the Central Arabian Sea and the Pacific Warm Pool

Diurnal variation in the air/sea heat flux balance leads to a diurnal cycle in sea surface temperature (SST) (Zhou *et al*. 2018, Shinoda 2005, Shinoda and Hendon 1998). This diurnal cycle (hereafter "the diurnal cycle") is crucial for understanding ocean phenomena such as the Tropical Warm Pool (Shinoda and Hendon, 1998) and the Madden-Julian Oscillation (MJO) (Bernie *et al*. 2005). This investigation uses the Price-Weller-Pinkel model (PWP) to simulate and explore the diurnal cycle at two locations in the tropics.

The following content will be covered:

1) Definition of the diurnal cycle; mechanisms; overview of PWP.
2) System of equations used; model variations
3) Datasets, resolution and initial conditions
4) Limitations and validation
5) Results and discussion
6) Conclusions

1 The Diurnal Cycle

The diurnal cycle is influenced by three key groups of fluxes (Figure 1):

1) Heat fluxes
   a. Sensible
   b. Latent
   c. Radiative (incoming solar and outgoing terrestrial)
2) Wind-induced momentum flux
3) Freshwater flux
   a. Precipitation
   b. Evaporation

In tropical waters, solar heating is strongly diurnal (Yang and Slingo, 2000). A thermal cycle is established when midday solar heating exceeds heat loss. In the absence of wind or other perturbation, the thermal response is limited to a thin surface layer (Price *et al*. 1986), due to the stabilising effects of thermal density stratification. Evaporation at the surface may increase salinity and thus density, weakening or reversing this stratification. Precipitation has the opposite effect. In the presence of winds, horizontal momentum imparted at the surface is transferred incompletely to lower levels by friction and entrainment, such that levels tend to have different horizontal velocities. This leads to shearing between levels, which favours vertical mixing, destabilising the column (Price *et al*., 1986; Swain, 2008). The balance between shear-induced destabilisation and thermal stabilisation is central to PWP.

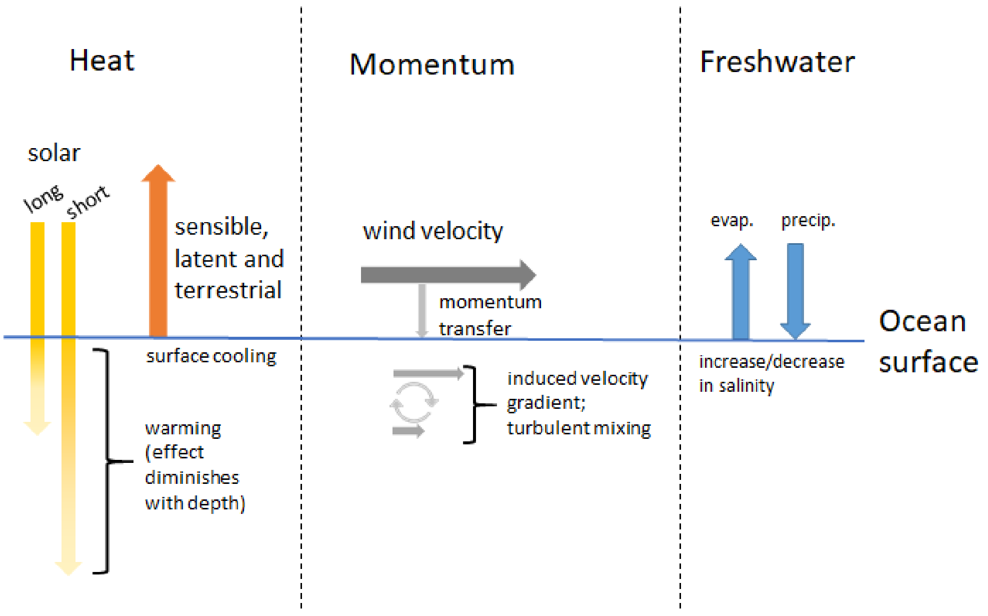

Heat
Momentum
Freshwater
solar
long
short
sensible, latent and terrestrial
wind velocity
momentum transfer
evap.
precip.
Ocean surface
surface cooling
warming (effect diminishes with depth)
induced velocity gradient; turbulent mixing
increase/decrease in salinity

Figure 1. Schematic of fluxes driving PWP. Arrow size not proportional to flux magnitude.

PWP starts with a one-dimensional vertical profile of a water column for which salinity, temperature and water velocity are known or interpolated at regular intervals. Each iteration of the model proceeds in the following way (Price *et al.* 1986):

1) Apply heat fluxes
    a. Apply latent and sensible heat fluxes to top level
    b. Absorb solar heat flux through the column, decreasing with a double exponential depth dependence
    c. Calculate new temperature profile
2) Apply freshwater flux
    a. Apply evaporation and precipitation to top level
    b. Recalculate salinity of top level
3) Convective mixing
    a. Calculate density profile of column from temperature and salinity profiles
    b. If density does not increase monotonically with depth, mix down until density stability is achieved, simulating free convection
4) Apply momentum flux evenly throughout mixed layer
5) "Bulk" mixing
    a. Check whole mixed layer (ML) for shear/buoyancy stability
    b. If instability is found, entrain successive levels of the subjacent stratified layer (SL) until stability is achieved
6) "Gradient" mixing
    a. Check SL for shear/buoyancy stability
    b. If instability is found, mix adjacent levels of greatest instability
    c. Repeat check/mix cycle until stability is achieved

2 Numerical method

The numerical method presented here was adapted from (Price *et al*. 1986). Steps that have been altered are marked with an asterisk. In step 4a, multiple methods were used. "Layer" refers to a defined depth region, such as ML or SL. "Level" refers to a profile section of thickness Δz and index j. See Appendix 7.1 for a glossary of symbols.

0) *Interpolate profile data to intervals of Δz using piecewise cubic Hermite interpolating polynomial.
1) Heat fluxes
    a. Sensible, latent, terrestrial: $T_{0,n+1} = T_{0,n} + \frac{Q_n}{\Delta z\, \rho_{0,n} C_p}\Delta t$
    b. Radiative:
        i. Calculate energy absorbed at each level: $I_z = I_0 \cdot [I_s e^{-\frac{z}{\lambda_s}} + I_l e^{-\frac{z}{\lambda_l}}]$
        ii. Use this in an equation analogous to (1a) for each level
2) Freshwater flux: $S_{0,n+1} = S_{0,n}(1 + \frac{W}{\Delta z}\Delta t)$
3) Convective mixing
    a. *Calculate density profile. Rather than the linear state equation used in (Price *et al*.. 1986), the Gibbs Seawater in situ density function was used.
    b. For any level k where $\rho_k < \rho_{k-1}$, mix levels 0 to k: $\{var\}_j = mean\{var\}_{0\ldots k}$ for all $0 \leq j \leq k$
4) Momentum flux
    a. Find base of ML:
        i. j_{bML} is first level where $\rho_j - \rho_0$ is greater than a predetermined threshold. ρ can be potential density (Price *et al*., 1978) or *density (Table 1).
        ii. *A temperature criterion can be used, such that j_{bML} is first level where $T_0 - T_j$ > threshold (Table 1).
    b. Depth of ML (MLD) = $j_{bML} \cdot \Delta z$
    c. $\Delta u_n = \frac{\tau_u}{MLD\, \bar{\rho}}\Delta t$ where $\bar{\rho}$ = average ρ for levels $0 \leq j \leq j_{bML}$
    d. $u_{j,n+1} = u_{j,n} + \Delta u_n$ for all levels $0 \leq j \leq j_{bML}$

    e. Repeat steps c-d for v
5) Bulk Richardson mixing
    a. Calculate R_b of ML:
    $R_b = \frac{g\,\delta\rho h}{\rho_r(\delta V)^2}$, δ{var} = $\{var\}_{j=j_{bML}+1} - \{var\}_{ML}$
    b. If $R_b < 0.65$, entrain subjacent level:
    $\{var\}_j = mean\{var\}_{0\ldots j_{bML}+1}$ for all $0 \le j \le j_{bML+1}+1$
    c. Repeat until $R_b \ge 0.65$
6) Gradient Richardson mixing
    a. Calculate R_g by first differences in SL:
    $R_g = \frac{g\delta\rho\Delta z}{\rho_r(\delta V)^2}$, δ{var} = $\{var\}_{j+1} - \{var\}_j$
    b. Check R_g values. If min(R_g) < 0.25, partially mix the two levels that have given rise to min(R_g):
        i. $\mu\{var\} = \frac{\{var\}_j - \{var\}_{j+1}}{2}$
        ii. d{var} = $\left(1 - \frac{R_g}{R_G}\right) \cdot \mu\{var\}$
        iii. $\{var\}'_j = \{var\}_j - d\{var\}; \{var\}'_{j+1} = \{var\}_{j+1} + d\{var\}$ where {var}' is the value after mixing
    c. Repeat check/mix cycle until all $R_g \ge 0.25$.

Run	MLD criterion	Threshold difference from surface value	Source
DPOT	potential density	0.125 kg m^{-3}	Monterey and Levitus, 1997
DD	density	0.1 kg m^{-3}	Dewey *et al.* 2017
DT02	temperature	0.2 °C	Thompson 1976
DT05	temperature	0.5 °C	Monterey and Levitus, 1997

Table 1: MLD criteria. (Price et al. 1978) has 0.2 sigma units as the potential density threshold criterion, while the Fortran code in (Price, undated) has 0.1 and states that "the degree of density homogeneity… is arbitrary and should not affect the results".

3 Datasets

Profile data was drawn from two sources: a year-long series of measurements at 15.5°N, 61.5°E in the Arabian Sea between October 1994 and October 1995 (AS) (Weller *et al.* 1998), and a five-month series at 1.75°S, 156°E gathered as part of the Tropical Ocean Global Atmosphere Coupled Ocean–Atmosphere Response Experiment (COARE) (Lagerloef *et al.*, 1997). In AS temperature trends are influenced by monsoonal seasonality (Zhou *et al.* 2018), while in COARE they are influenced by the MJO (Bernie *et al.* 2005). Datasets include hourly measurements of salinity (S), temperature (T), and meridional and zonal water velocity (UV). Defining a full profile to be one for which data is available for S, T and UV, full profile depth was 124 m in COARE and 250 m in AS. Vertical resolution to 100 m is shown in Figure 2. Measurements were interpolated to 1 m intervals, following (Dewey *et al.* 2017). Interpolated profiles provided initial model conditions.

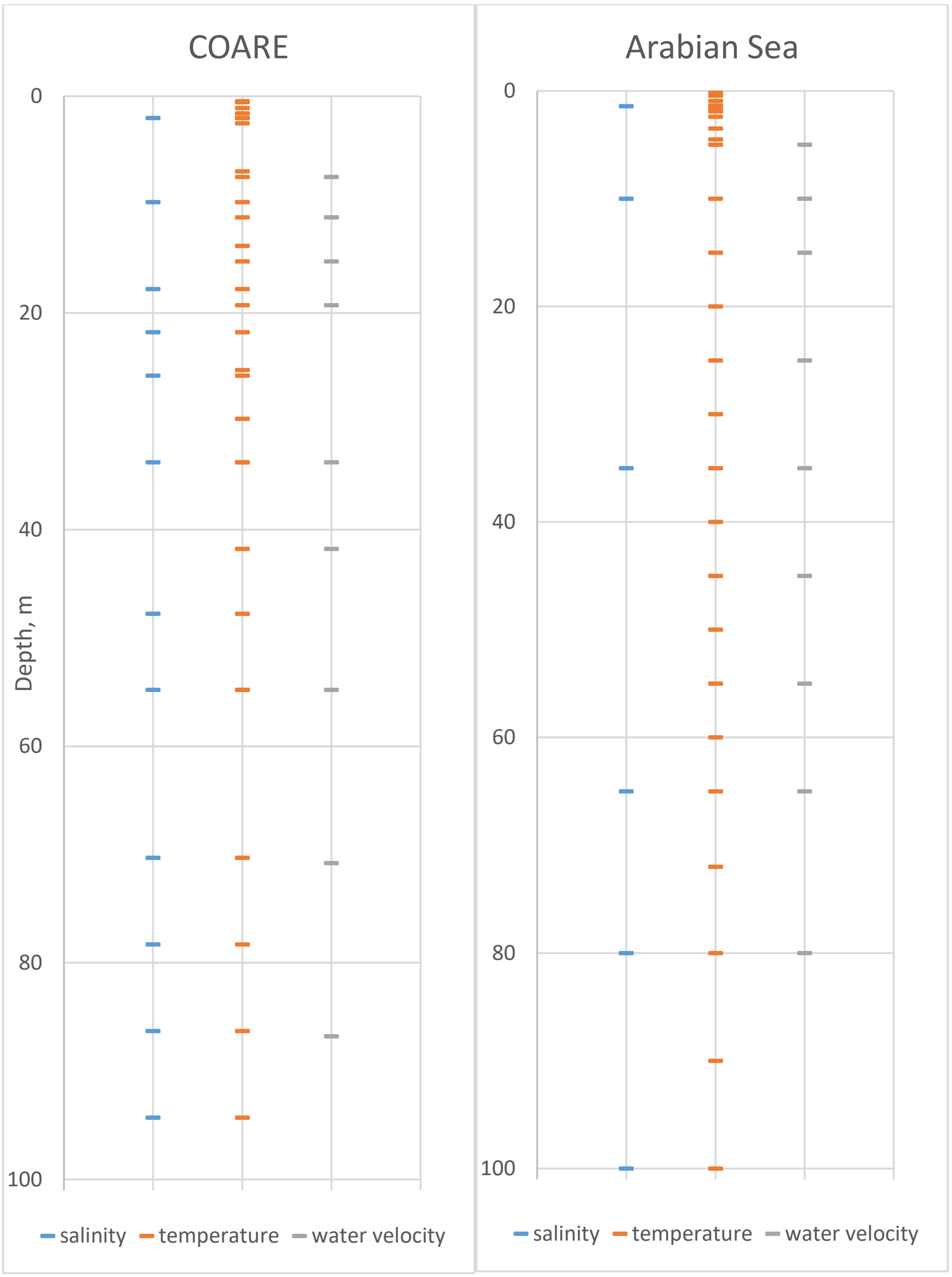


Figure 2: Vertical resolution of measurements in upper 100 m

Datasets also contained hourly measurements of heat fluxes (solar, terrestrial, latent and sensible) and zonal and meridional wind stress. Hourly precipitation and evaporation data were taken from ECMWF ERA5 (Hersbach *et al*. 2018).

4 Model limitations and validation

4.1 Testing, development and limitations

During preliminary model testing, difficulty was found in attaining SL stability (step 6) from initial profiles. This was resolved by setting initial water velocity to 0 ms^{-1} throughout the column. During development, modelled and observed profiles were plotted at each timestep, in conjunction with readouts of MLD and mixings being conducted (see example in Appendix 7.2). This allowed for visual step-by-step validation, sanity checking, and localisation of error sources. The following issues were found:

4.1.1 MLD criterion impact

The different possible criteria by which to determine MLD gave values that differed by a factor of 2, as a result of which it was decided to test accuracy of diurnal cycle magnitude (DCM) and SST for each criterion.

4.1.2 Bulk overmixing

During bulk mixing the model would occasionally entrain up to 80 levels at one iteration. Although this occurred in <0.1% of iterations, and did not have a lasting impact on DCM, it led to a rapid deepening of the ML and cooling of SST that was inconsistent with observations. This overmixing was resolved by adding a "nudge" subroutine to the bulk mixing step: when a certain number of levels had been entrained at one iteration, the critical Richardson value R_{b_crit} was temporarily lowered from 0.65 to 0.5. The optimum number of levels after which to nudge was found to be 28, to minimise both overmixing and unnecessary application of the nudge subroutine.

4.1.3 Gradient looping

Model step 6, which carries out gradient Richardson mixing, occasionally failed to converge, if stability was slow to emerge. The number of gradient stability check/mix cycles was therefore limited to a "gradmax" of 1000 per model iteration.

4.1.4 Review of limitations

Records were kept of the number of times that each corrective measure was applied. On control runs, bulk nudging typically occurred once in a year of data, and gradient maxing 2-3 times. During sensitivity testing these values could be higher. Runs that had anomalously high frequency of nudges or gradmaxes will be highlighted. Validation against observations (see below) suggested that none of these adjustments introduced problematic errors in DCM, and thus that the model as coded is suitable for the purposes of this investigation. Nonetheless, the overmixings indicate sources of instability that would need to be addressed before application to other datasets.

4.2 Validation

Validation of three types will be presented:

1) Visual comparison of SST and DCM graphs for observations and model runs
2) Visual comparison of contour graphs of temperature profile evolution
3) Statistical measures

4.2.1 SST and DCM

As MLD criterion runs were validated independently this section will also explain how final criteria were selected.

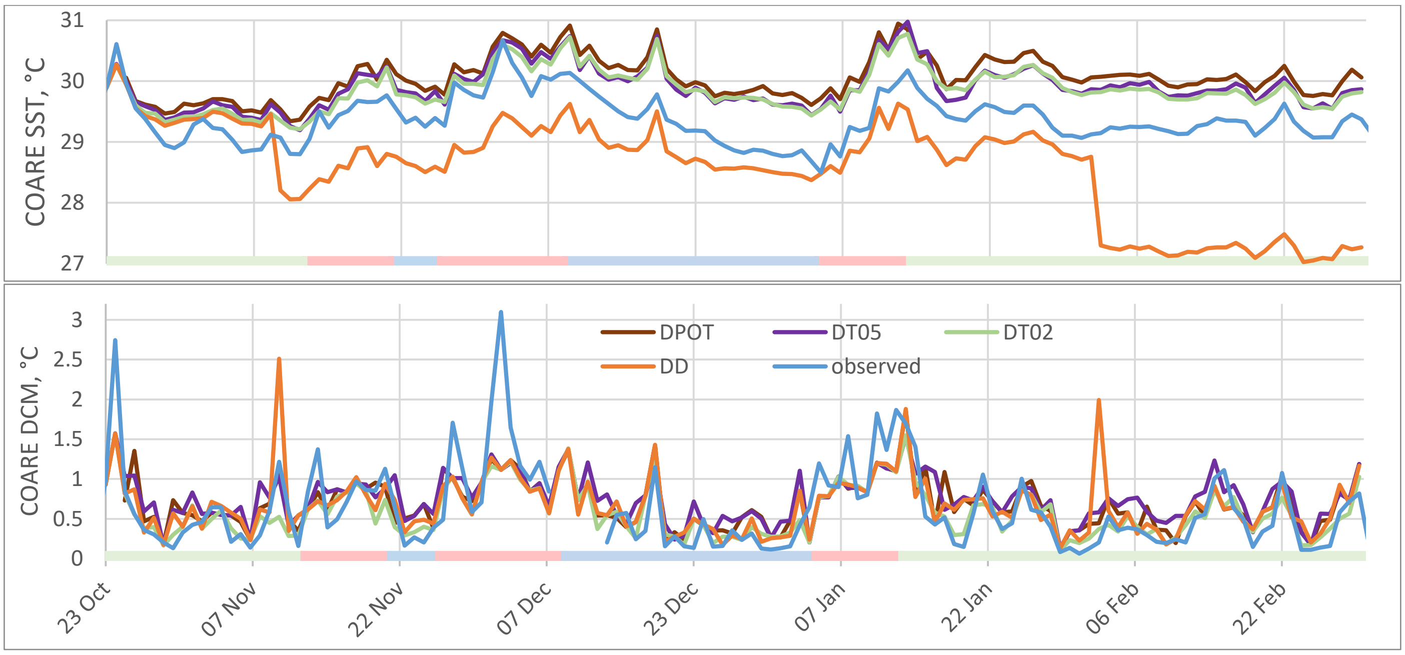


Figure 3: modelled SST and DCM for COARE dataset. Coloured bars along x axes represent periods of warming (red), cooling (blue) or neutral temperature trend. Warming and cooling trends are associated with suppressed and active phases of the Madden-Julian Oscillation, respectively. Colour schemes for model runs are consistent between graphs. There is a gap in observed SST between 9/12/92 and 13/12/92 inclusive. This has been interpolated for graphing. For SST results compare (Bernie et al. 2005) Figure 1

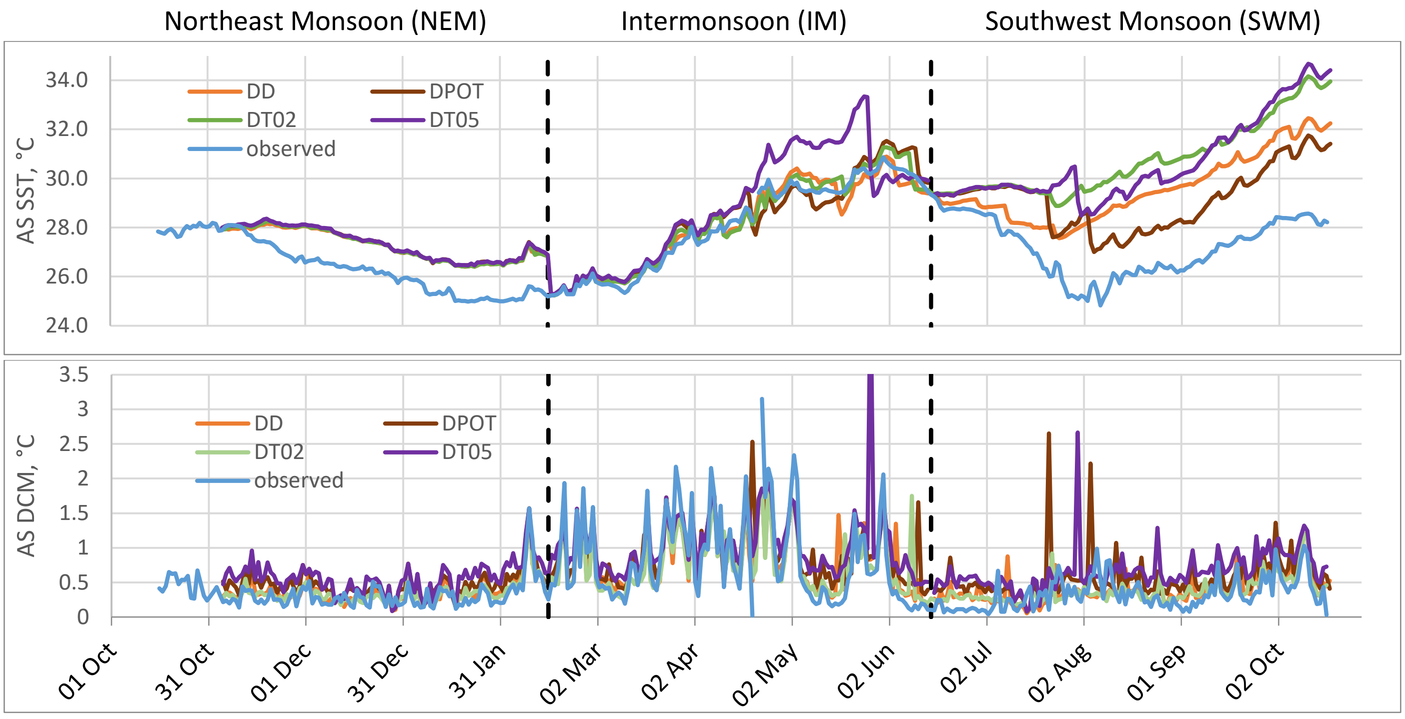


Figure 4: modelled SST and DCM for AS dataset. Following (Zhou et al., 2018), all models were reinitialised at season transitions (marked by vertical dashed lines). Compare (Zhou et al. 2018) Figure 2c,d.

In COARE (Figure 3), all model runs track observed SST with visually similar levels of accuracy, with the exception of DD, in which rapid cooling results from overmixing on 10 Nov and 3 Feb. These events are associated with spikes in DCM, but DCM then continues to track observed values. DCM tends to be higher during suppressed phases of the MJO in all runs, consistent with the findings of (Bernie *et al*. 2005). In AS (Figure 4), all SST outputs are consistent with observations during NEM. Only DT05 clearly deviates during IM, while all runs warm excessively in SWM. This divergence coincides with the horizontal advection of a cool filament towards the study site (Fischer *et al*., 2002, discussed in Zhou *et al*., 2018). PWP, being a 1D vertical model, is unable to reproduce this. DCM results are broadly consistent with observations.

4.2.2 Temperature profile evolution

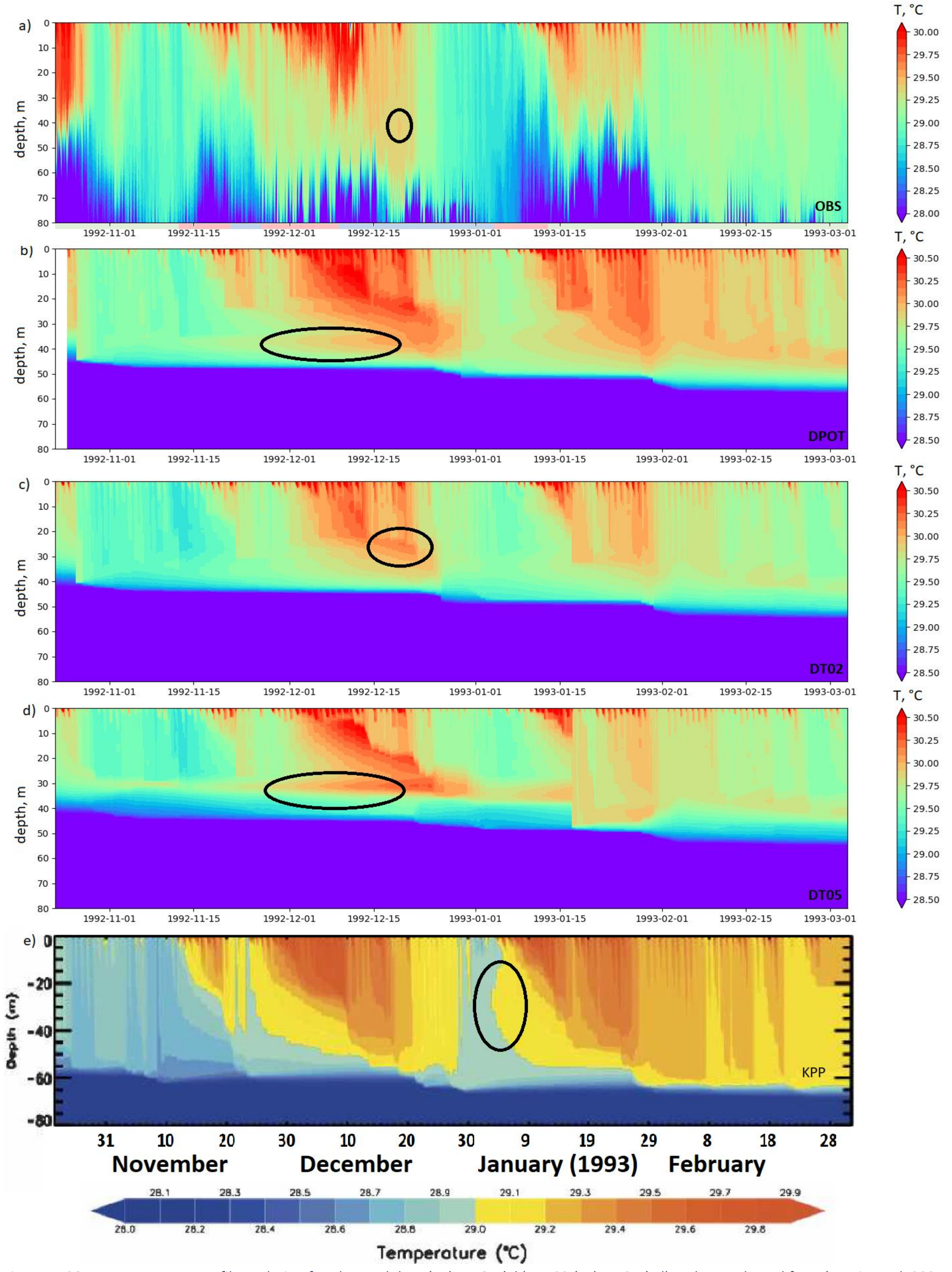


Figure 5: COARE temperature profile evolution for observed data (5a), DPOT (5b), DT02 (5c), DT05 (5d) and KPP adapted from (Bernie et al. 2005, fig. 2) (5e). "Heat islands" ringed in black. Temperature scale is 0.5 °C higher for modelled profiles than observed to compensate for drift and aid visual comparison, except in (6e) where the same effect is achieved by subtracting 0.5 °C from the values. Month labels in (5e) have been clarified.

In COARE, the overmixing in DD led to uninformative temperature profiles, which are therefore not shown here (see Appendix 7.3.1). Of the other runs, DT02 appears the best match, particularly because it has fewest areas of isolated, warm water at depth (Figure 5). These "heat islands" are not entirely absent from the observed data, but are much smaller and less frequent than in the model runs. Comparison of DT02, OBS and the control run of (Bernie *et al*., 2005)'s K-Profile Parameterisation (KPP) model (Figure 5e) suggests that a broadly similar level of accuracy has been achieved.

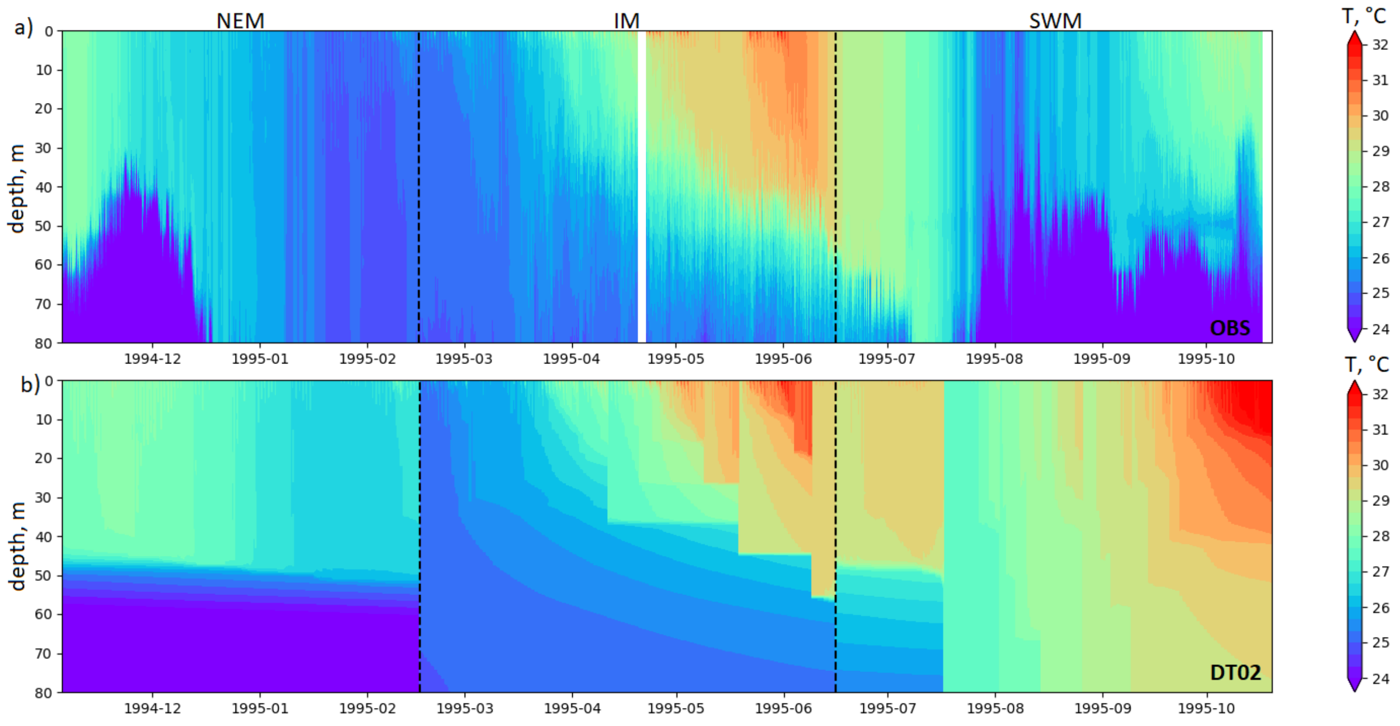


Figure 6: AS temperature profile evolution for observed data (7a) and DT02 (7b).

In AS, outputs were more consistent between MLD criterion runs, so comparison between them is less instructive and only one is shown here (Figure 6; remaining plots in Appendix 7.3.2). Modelled profiles are broadly consistent with observations during NEM and closely so during IM. Outputs diverge from observations during SWM, as expected from SST trends and the abovementioned horizontal advection event.

4.2.3 Statistical measures

Root mean square error (RMSE) was used to compare modelled DCM, MLD and SST with observed values for all model runs. RMSE is a standard measure to assess model accuracy against observed values (Armstrong and Collopy, 1992), aggregating errors and penalising outliers (Pontius *et al*., 2007). For DCM and MLD, RMSE was normalised (nRMSE = RMSE/observed mean) to facilitate comparison between runs and segments. Pearson's R, a measure of correlation strength, was added as a complementary indicator of DCM accuracy because (n)RMSE quantifies error but ignores trend. Mean and standard deviation (σ) of modelled DCM, MLD and SST were also compared to observed values. As detailed discussion of all of these measures is beyond the scope of this report, and the primary intention was to assess diurnal cycle simulation accuracy, analysis will focus on DCM (Figure 7). For full data see Appendix 7.4.

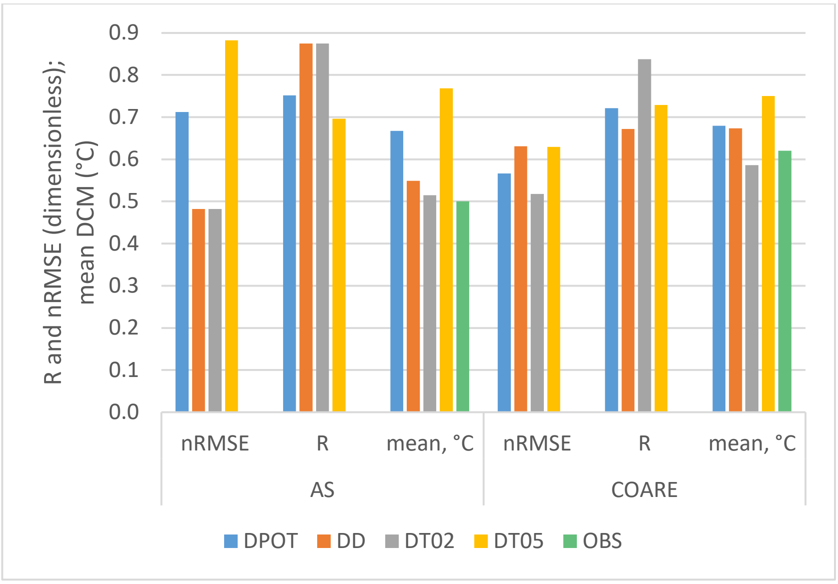


Figure 7: nRMSE and R of modelled against observed DCM, and mean DCM values for comparison with observed mean.

In AS, DD and DT02 have highest R and lowest nRMSE, indicating highest accuracy. This was consistent across all seasons (segmented data in Appendix 7.4.2). In COARE, DT02 performs best, while DD performs least well, though the margins here are smaller. There was more inter-phase variability in COARE than in AS, but on both measures of accuracy and in all phases either DT02 or DD was the frontrunner (Appendix 7.4.3). As no single criterion appeared superior on all counts, subsequent sensitivity tests were carried out with both DD and DT02.

5 Results and discussion

Non-control model runs fall into three categories:

1) Model variation
2) Forcing redistribution
3) Forcing alteration

5.1 Model variation

Two types of model variation were undertaken. MLD criterion variation has been discussed in validation, above. The second type was running the model with only one type of mixing: convective, bulk or gradient (Figure 8).

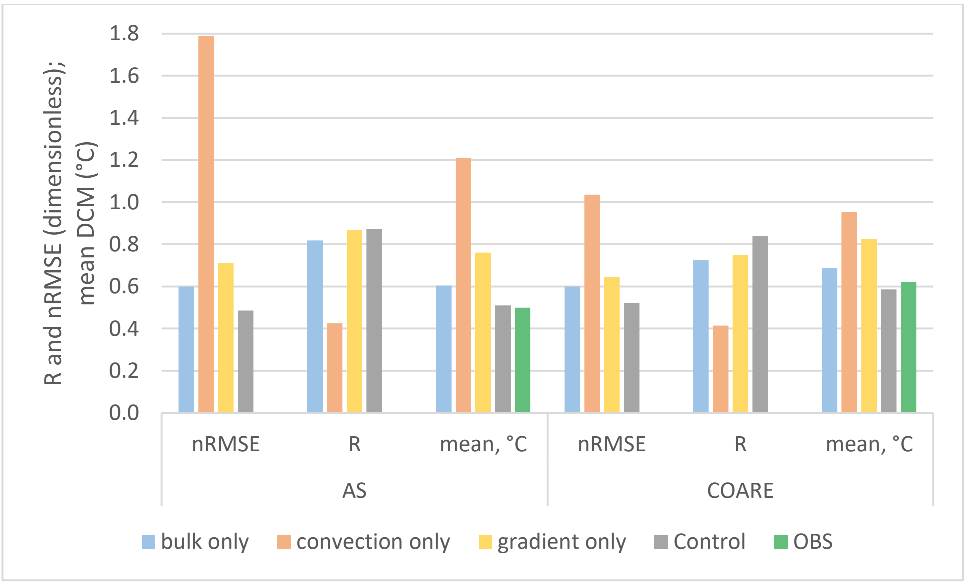


Figure 8: comparison of DCM values and statistics for model variations.

For these tests only DT02 was used. Bulk-only and gradient-only perform similarly to Control in terms of correlation, but nRMSE rises. For convection-only, losses of accuracy are much more pronounced. This is consistent with the finding in (Price *et al.*, 1986) that convection mixing makes the least contribution to the model. Frequency of bulk nudge and gradmax adjustments was high here, being applied to 5.9 % and 7.7 % of timesteps respectively.

5.2 Forcing redistribution

5.2.1 Fixed means

For these runs, heat and wind forcing values were set to mean values for a specified period. For 24HR, this was midnight to midnight, local time. For 12HR, 12 hour periods centred on noon and midnight were used. For 12HR_off the periods were offset by 6 hours to centre on 0600 and 1800. 12HR reduces the amplitude of the diurnal heat flux cycle; 12HR_off reduces it further and 24HR eliminates it (Figure 9; equivalent for AS, showing similar patterns, in Appendix 7.5). This is reflected in progressively lower mean DCM values (Figure 10; segmented data in Appendix 7.6 show consistency of pattern), highlighting the dependence of DCM on the solar-forced diurnal heat flux cycle.

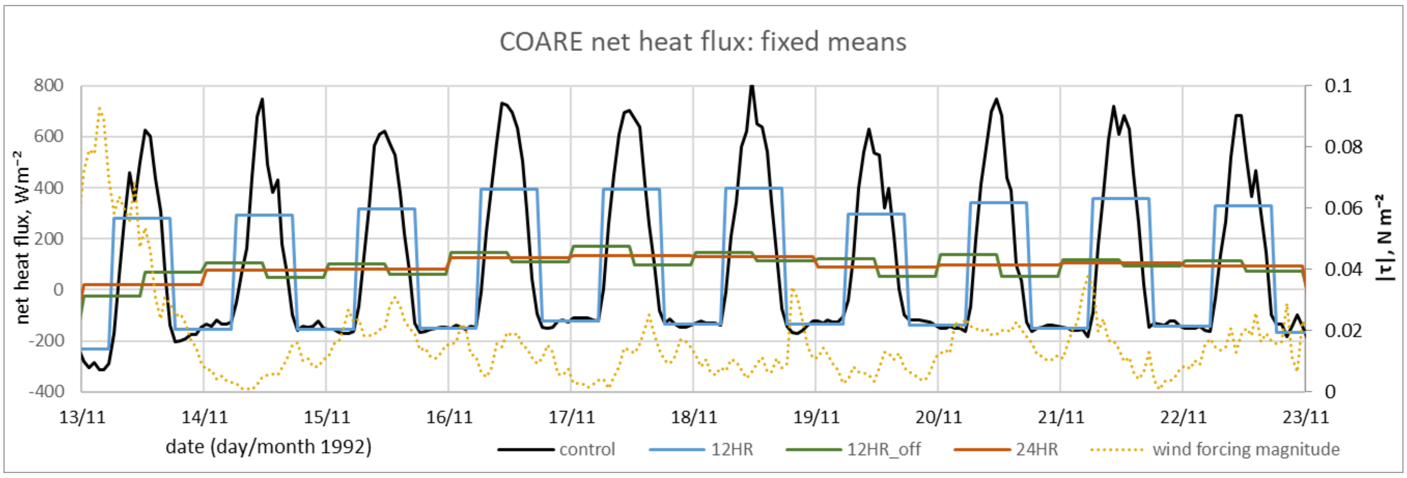


Figure 9: Representative 10 day period showing net heat forcing for each fixed mean run, COARE. Unaveraged momentum forcing magnitude also shown to illustrate lack of diurnality.

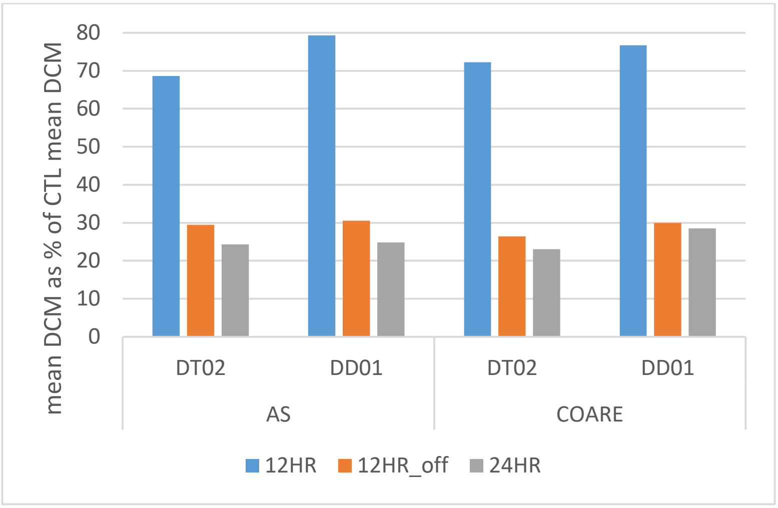


Figure 10: mean DCM as percentage of mean control DCM when model is forced with fixed mean forcings for each period type, dataset and MLD criterion. Compare similar but more pronounced results from KPP model in (Bernie et al. 2005), Table 1.

5.2.2 Trailing means

To separate the influence of momentum and heat forcings, the model was run on forcings with a 24 hour trailing mean smoothing applied to heat (24Q), momentum (24τ) or both sets of forcings (24Qτ) (Table 2).

	Smoothing applied	COARE		AS	
		DT02	DD	DT02	DD
Mean DCM as % of CTL	24Q	25%	30%	28%	27%
	24τ	89%	94%	95%	95%
	24Qτ	25%	29%	30%	25%

Table 2: as Table 2, for trailing mean model runs.

24Q reduces mean DCM to 25-30 % of control values, as with the fixed mean runs. 24τ has a far smaller impact, while 24Qτ percentages are similar to 24Q. This suggests that DCM is more strongly influenced by diurnality of heat fluxes than momentum fluxes. DT02 24Qτ had the highest adjustment frequency of any run, with gradmax applied to 8.9 % of timesteps (4.9 % in DD). Results were nonetheless consistent with expectations based on 24Q and 24τ.

5.3 Forcing alteration

5.3.1 Freshwater alteration

Setting all FW forcings to 0 for run noFW had limited impact on nRMSE, R and mean DCM (Figure 11).

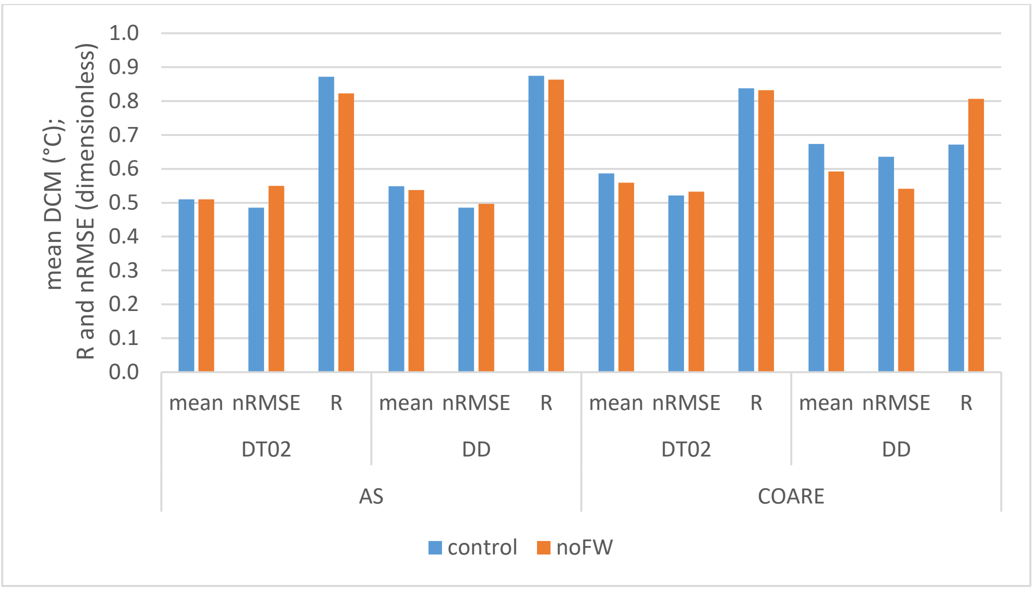


Figure 11: DCM mean and statistical measures of control runs and noFW. Note increase in R and decrease in nRMSE for DD noFW in COARE.

The largest difference between the control and noFW runs is an improvement in measured accuracy in DD COARE. As FW influences salinity, and thus density, it is intelligible that noFW differs from CTL more in DD than DT02. The improvement in accuracy may be explained by spatial heterogeneity of the precipitation field (Bernie *et al*., 2005), meaning that including FW merely adds noise to the data. That this effect is absent from AS may be explained by the lower mean and σ of FW forcings in that dataset (Figure 12). Lower σ indicates lower variability, and thus reduced capacity to add noise to the model if data are inaccurate.

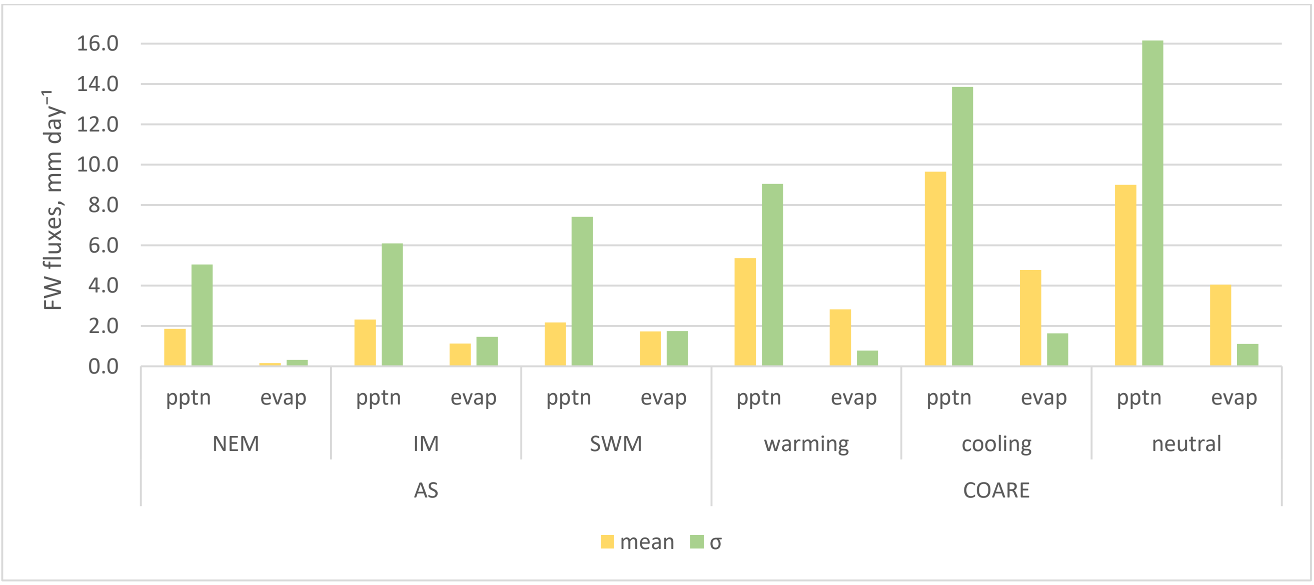


Figure 12: mean and standard deviation of FW forcings. Hourly ERA 5 data, converted to mm/day. AS data is segmented by season, COARE data by warming/cooling trend.

The higher standard deviation of precipitation in neutral periods of COARE might suggest that the gains in accuracy should be highest there, and this is found to be the case (Figure 13). There is a drop in R during cooling periods, which also have high mean freshwater fluxes (Figure 12), but nRMSE improves slightly. Overall this suggests that FW has little impact on DCM when FW forcings are as low as in AS, and that at higher values it may be counterproductive to include those forcings when data are of the quality and resolution used here.

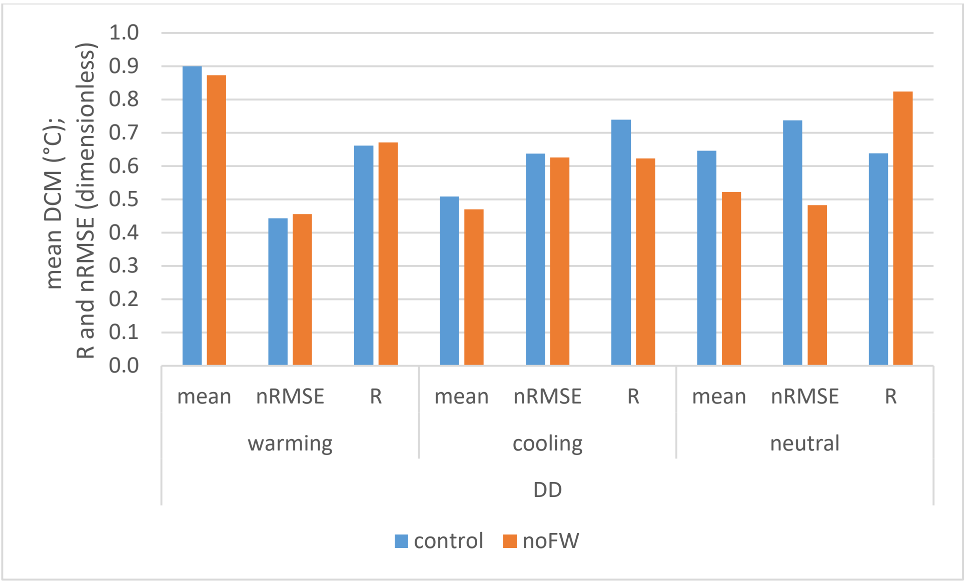


Figure 13: DCM mean and statistical measures of control and noFW for COARE with density MLD criterion

5.3.2 Momentum flux alteration

Bulk and gradient mixing are wind-driven (Price *et al*., 1986). Strong winds enhance turbulent mixing and entrainment, deepening the ML, spreading heat deeper in the column, and thus reducing DCM (Mukherjee and Tandon, 2016). Runs ½τ and 2xτ set out to verify this. In both AS and COARE, ½τ resulted in shallower ML and greater DCM, while the opposite was the case for 2xτ (Table 3), consistent with expectations. This pattern obtained in all seasons (AS) and MJO phases (COARE) for which full data was available (segmented results in Appendix 7.7).

Run	AS		COARE	
	MLD, m	DCM, °C	MLD, m	DCM, °C
CTL	28.3	0.51	27.6	0.59
½τ	22.4	0.63	21.4	0.68
2xτ	42.6	0.43	50.8	0.54

Table 3: ½τ and 2xτ results (DT02 only)

6 Conclusions

Validation against observations has shown that PWP as coded here adequately simulates the upper ocean diurnal cycle in the tropics in both regions and all phases/seasons examined, except where horizontal advection plays a substantial role. Statistical analysis and comparison with other published model outputs suggest an acceptable level of accuracy has been attained.

Results presented indicate that thermal flux diurnality is instrumental in setting up the diurnal cycle, and that greater momentum forcing reduces DCM, while freshwater forcings have limited impact. Convective mixing appeared to be the least important of the three mixing processes. These findings are consistent with the literature reviewed.

References

Armstrong, J.S. and Collopy, F., 1992. Error measures for generalizing about forecasting methods: Empirical comparisons. *International Journal of Forecasting,* **8**, 69-80

Bernie D.J., Woolnough, S.J. and J. M. Slingo, 2005. Modeling diurnal and intraseasonal variability of the ocean mixed layer. *Journal of Climate*, **18**(8), 1190-1202

Dewey, S.R., Morison, J.H. and Zhang, J., 2017. An edge-referenced surface fresh layer in the Beaufort Sea seasonal ice zone. *Journal of Physical Oceanography*, **47**(5), 1125-1144.

Fischer, A. S., Weller, R. A., Rudnick, D. L., Eriksen, C. C., Lee, C. M., Brink, K. H., et al. (2002). Mesoscale eddies, coastal upwelling, and the upper-ocean heat budget in the Arabian Sea. *Deep Sea Research Part II: Topical Studies in Oceanography*, **49**(12), 2231-2264.

Hersbach, H., Bell, B., Berrisford, P., Biavati, G., Horányi, A., Muñoz Sabater, J., Nicolas, J., Peubey, C., Radu, R., Rozum, I., Schepers, D., Simmons, A., Soci, C., Dee, D., and Thépaut, J-N, 2018. ERA5 hourly data on single levels from 1979 to present. Copernicus Climate Change Service (C3S) Climate Data Store (CDS). 10.24381/cds.adbb2d47

Lagerloef, G.S.E., Lukas, R., Weller, R.A. and Anderson, S.P., 1998. Pacific warm pool temperature regulation during TOGA COARE: Upper ocean feedback. *Journal of Climate*, **11**(9), 2297-2309

Monterey, G. and Levitus, S., 1997. *Seasonal Variability of Mixed Layer Depth for the World Ocean*. NOAA Atlas NESDIS 14, U.S. Government Printing Office, Washington D.C.

Mukherjee, S. and Tandon, A., 2016. Comparison of the simulated upper-ocean vertical structure using 1-dimensional mixed-layer models. *Ocean Science Discussions*, 1-22

Pontius, R. G., Thontteh, O., and Chen, H., 2007. Components of information for multiple resolution comparison between maps that share a real variable. *Environmental and Ecological Statistics* **15**(2), 111-142

Price, J.F., undated. *Fortran version of the PWP model. Minor, minor changes from the original, 1986 version*. Available online at https://www.whoi.edu/science/PO/people/jprice/PWP/pwp.f. Last accessed 27/12/2020

Price, J.F., Mooers, C.N.K. and Leer Van, J.C., 1978. Observation and simulation of storm-induced mixed-layer deepening. *Journal of Physical Oceanography*, **8**(4), 582-599

Price, J.F., Weller, R.A., and Pinkel, R., 1986. Diurnal cycling: observations and models of the upper ocean response to diurnal heating, cooling, and wind mixing. *Journal of Geophysical Research*, **91**(C7), 8411-8427

Shinoda, T., 2005. Impact of the diurnal cycle of solar radiation on intraseasonal SST variability in the western equatorial Pacific. Journal of Climate, **18**(14), 2628–2636.

Shinoda, T. and Hendon, H. H., 1998. Mixed layer modelling of intraseasonal variability in the tropical Western Pacific and Indian Oceans. *Journal of Climate*, **11**(10), 2668-2685

Swain, D., 2008. *Prediction of mixed layer depth for ocean state forecast*. PhD Thesis, Savitribai Phule Pune University, Pune. Available online at http://hdl.handle.net/10603/126025

Thompson, R.O.R.Y, 1976. Climatological numerical models of the surface mixed layer of the ocean. *Journal of Physical Oceanography*, **6**(4), 496-503

Weller, R.A., Baumgartner, M.F., Josey, S.A., Fischer, A.S. and Kindle, J.C., 1998. Atmospheric forcing in the Arabian Sea during 1994-1995: observations and comparisons with climatology and models. *Deep Sea Research II- Topical studies in oceanography*, **45**(10-11), 1961-1999

Yang, G. and Slingo, J., 2000. The diurnal cycle in the tropics. *Monthly Weather Review*, **129**(4), p784-801

Zhou, S., Zhai, X. and Renfrew, I. A., 2018. The impact of high-frequency weather systems on SST and surface mixed layer in the central Arabian Sea. *Journal of Geophysical Research: Oceans*, **123**(2), 1091-1104

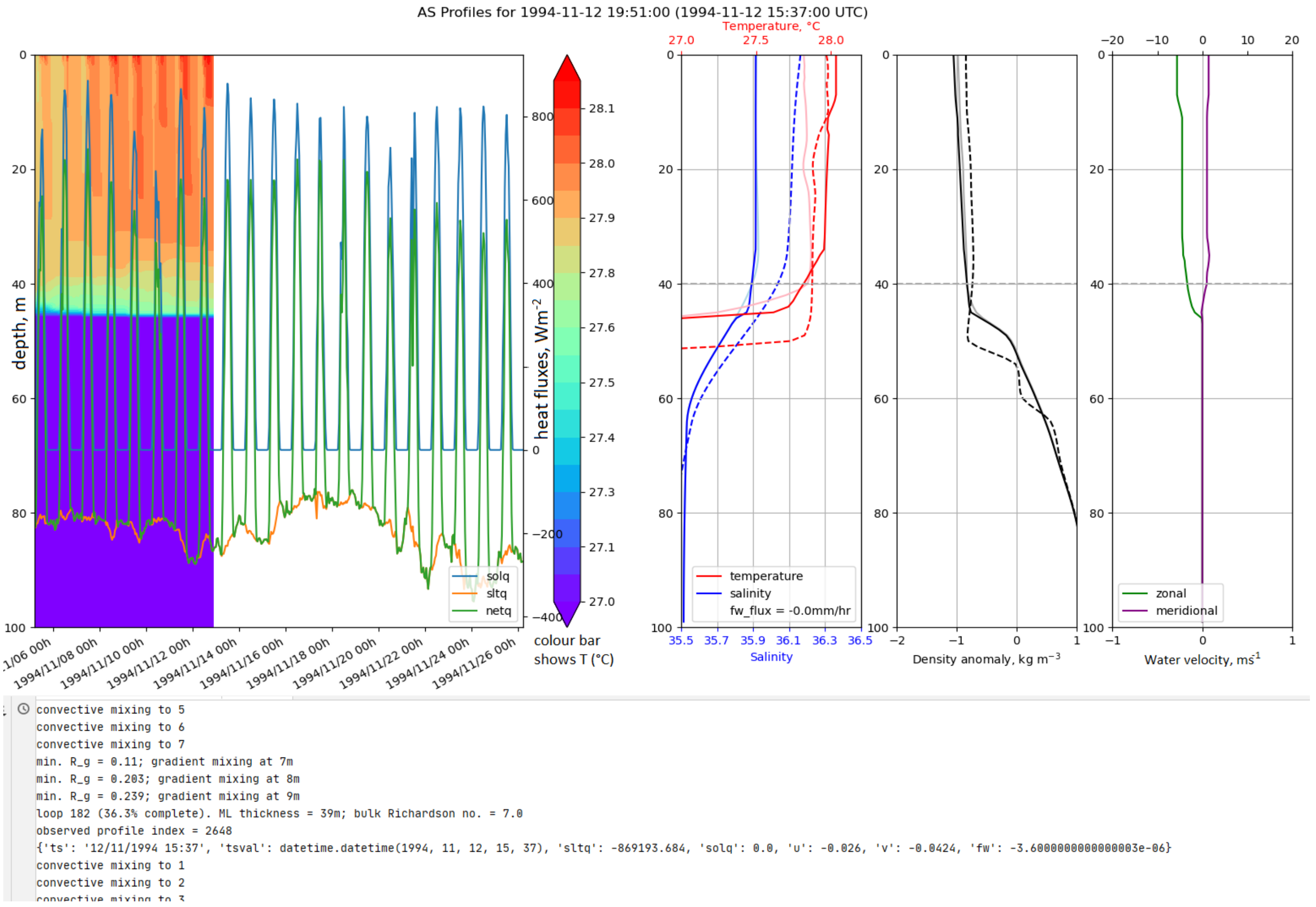
7 Appendices

7.1 Symbols used, with units

z	depth, increasing downward, m
j	level index, increasing downward
{var}	any variable
Δ{var}	increment of {var}
$\{var\}_z$	{var} at depth z
$\{var\}_j$	{var} at level j
$\{var\}_n$	{var} at timestep n
T	temperature, K
Q	sum of net latent and sensible heat fluxes into ocean, W m^{-2}
S	salinity, ppt
W	freshwater flux, evaporation minus precipitation, m s^{-1}
C_p	specific heat capacity of water, J K^{-1} kg^{-1}
I_z	solar energy absorbed at depth level z, W m^{-2}
$I_{s,l}$	shortwave,longwave insolation fraction, dimensionless
$\lambda_{s,l}$	empirically determined absorption terms, m
ρ_r, T_r, S_r	reference values for ρ, T, S used in salinity and stability calculations
α, β	constants of proportionality for density calculations (units evident from context)
u,v	zonal,meridional current velocity, m s^{-1}
$\tau_{u,v}$	zonal,meridional wind stress
R_b	bulk Richardson number
R_g	gradient Richardson number
R_G	constant slightly larger than critical R_g used to hasten convergence

Return to 2: Numerical method

7.2 Diagnostic testing during development process



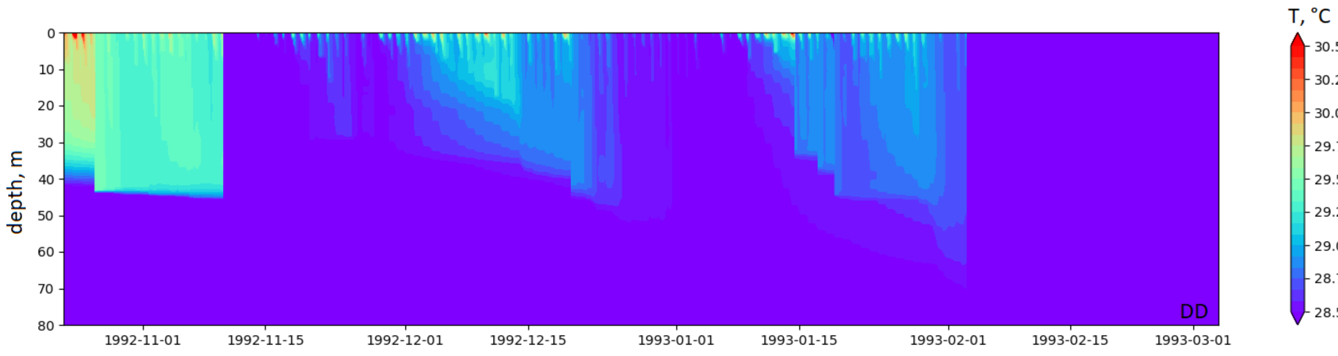
```
convective mixing to 5
convective mixing to 6
convective mixing to 7
min. R_g = 0.11; gradient mixing at 7m
min. R_g = 0.203; gradient mixing at 8m
min. R_g = 0.239; gradient mixing at 9m
loop 182 (36.3% complete). ML thickness = 39m; bulk Richardson no. = 7.0
observed profile index = 2648
{'ts': '12/11/1994 15:37', 'tsval': datetime.datetime(1994, 11, 12, 15, 37), 'sltq': -869193.684, 'solq': 0.0, 'u': -0.026, 'v': -0.0424, 'fw': -3.6000000000000003e-06}
convective mixing to 1
convective mixing to 2
convective mixing to 3
```

Figure 14: Example of visual diagnostic testing during model development. On the contour graph, fluxes shown are solar (solq), summed sensible, latent and terrestrial (sltq), and net (netq). On the profile plots, pale lines show initial values, solid lines show modelled values, and dashed lines show observed values. Included as clarification of development method, not presentation of results.

Figure 14 Shows a screenshot of the visual diagnostic testing process. During development, the incomplete model could be observed in real time as it iterated, plotting observed and modelled data simultaneously. Overmixing events were easily identified by the rapid straightening of the temperature profile and homogenising of the contour plot. The plausibility of heat penetration could be checked by correlation with heat fluxes. Heat islands were found to be consistent with the working of the model as they were compensated by low salinity values, allowing density to increase monotonically despite increasing temperature.
Return to 4.1: Testing, development and limitations

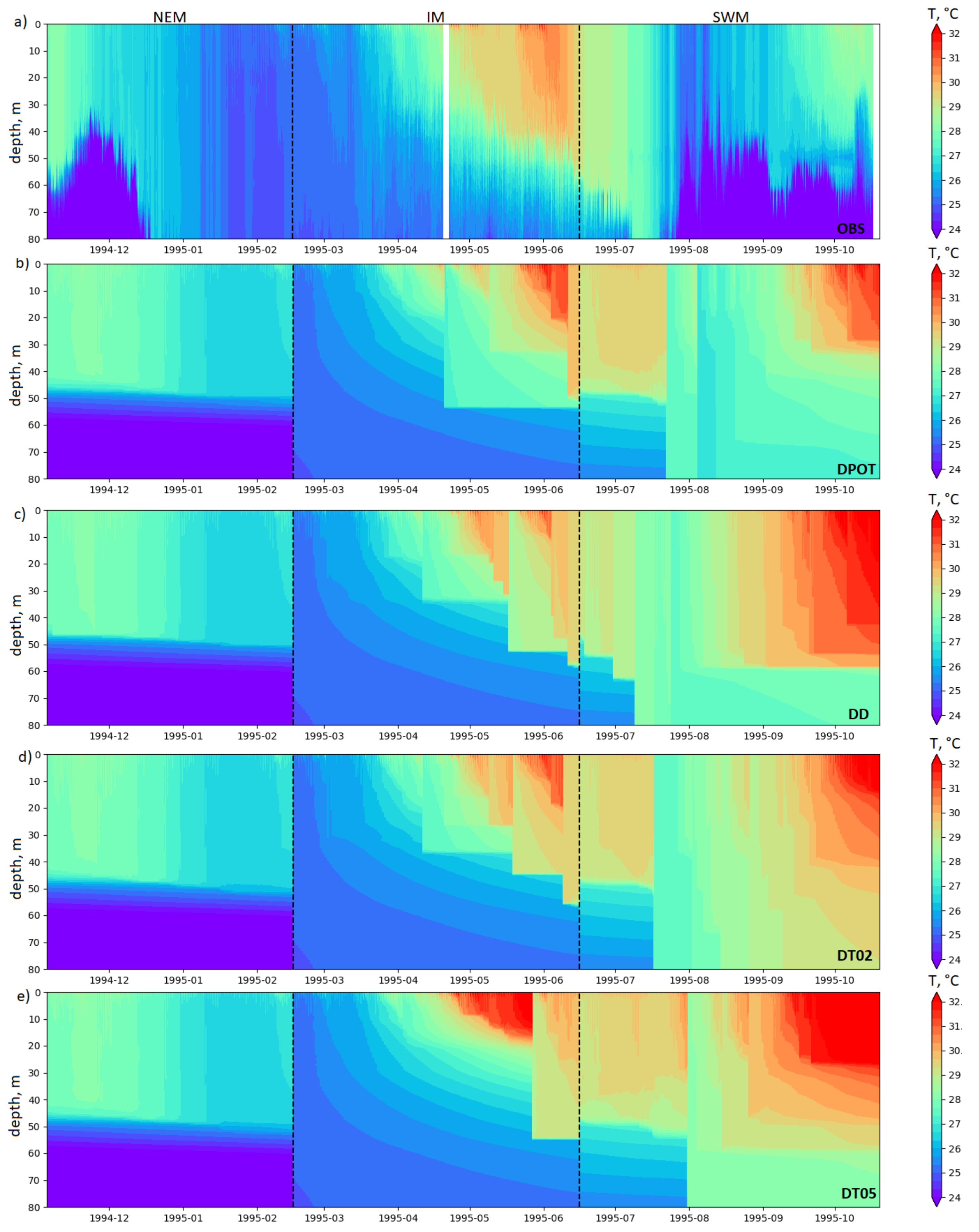
7.3 Temperature profile evolution contour plots not included in main text

7.3.1 COARE



Due to an overmixing event around 8/11/92 this plot does not provide a visually informative comparison for the other MLD criterion plots. If comparing COARE plots in this report to the figures in (Bernie *et al.*, 2005), note that the month labels there appear to be offset. For example, "February" 1993 is written below a month that has 29 days, and "December" follows a month that has 31. Actual dates can be confirmed from comments in the text.
Return to 4.2.2: Temperature profile evolution

7.3.2 AS



AS contour outputs were broadly consistent across all MLD criteria. Divergence between model runs is greater during SWM, but all models diverge from observed values here for reasons discussed in the report, making inter-model comparisons less meaningful.

Return to Figure 6: AS temperature profile evolution for observed data (7a) and DT02 (7b).

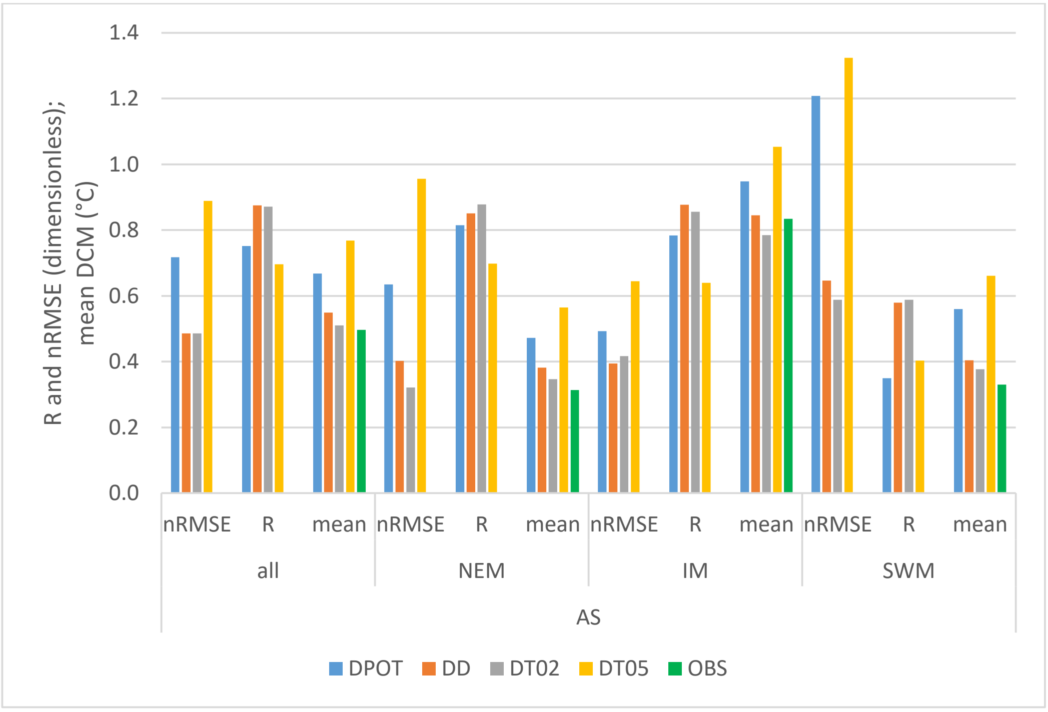
7.4 Detailed statistical model comparison data

7.4.1 Summary of MLD, SST and DCM accuracy for each MLD criterion

		MLD	SST, °C	DCM	
Dataset	MLD criterion	nRMSE	RMSE	nRMSE	R
AS	DPOT	0.56	1.46	0.71	0.75
	DD	0.38	1.77	0.48	0.87
	DT02	0.56	1.64	0.48	0.87
	DT05	0.55	2.48	0.88	0.70
COARE	DPOT	0.44	0.71	0.57	0.72
	DD	0.21	1.08	0.63	0.67
	DT02	0.47	0.53	0.52	0.84
	DT05	0.45	0.57	0.63	0.73

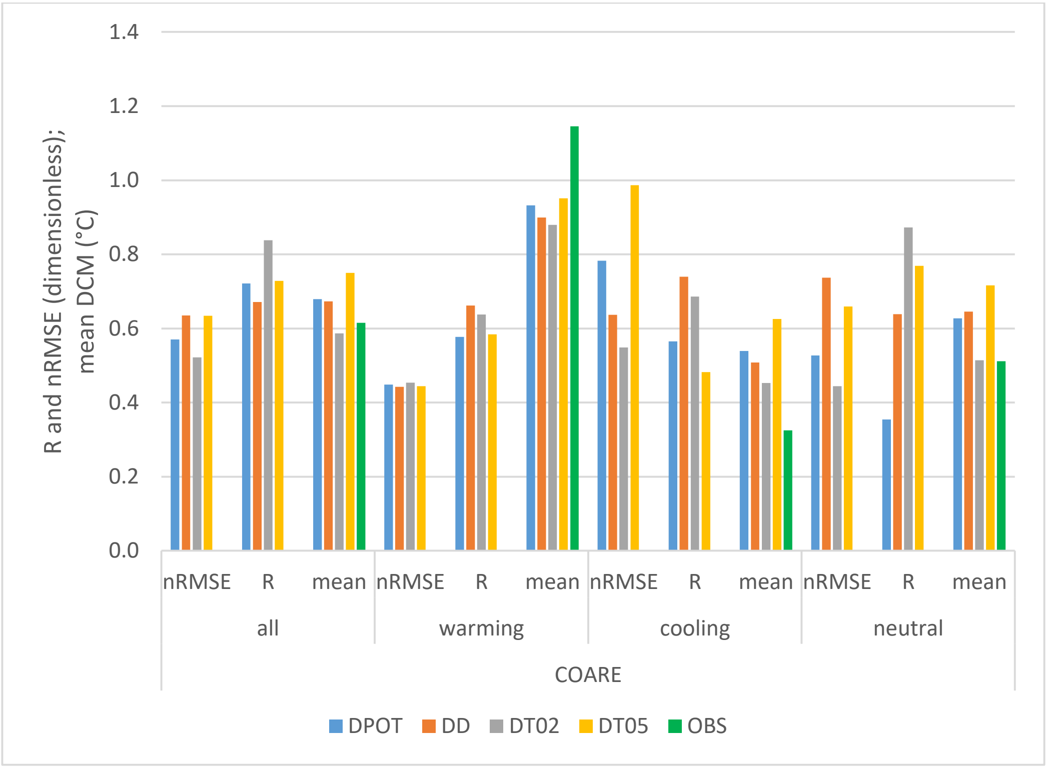
Darker green indicates higher accuracy. In AS, DD and DT02 are clear frontrunners. In COARE, DT02 is superior on most counts, but DD models MLD much better than any other criterion does, arguing for its inclusion in further tests.
Return to 4.2.3: Statistical measures

7.4.2 Segmented DCM accuracy data, AS



In all seasons, DD and DT02 have the lowest nRMSE and the highest R. Mean DCM is also consistently closer to the observed value for these two runs.
Return to Figure 7: nRMSE and R of modelled against observed DCM, and mean DCM values for comparison with observed mean.

7.4.3 Segmented DCM accuracy data, COARE



In all phases and on both measures of accuracy, either DD or DT02 is the frontrunner, but the pattern is not as clear as in AS. For example, in neutral MJO phases, DD has the highest nRMSE, despite having the lowest in cooling phases.

Return to Figure 7: nRMSE and R of modelled against observed DCM, and mean DCM values for comparison with observed mean.

7.4.4 Full model statistics for all runs, AS dataset

The tables presented below (Figure 15, Figure 16) contain full statistical data for all runs, including some not used in the body of the report. Runs beginning "CC", for example, used convection mixing only, with a constant mixing depth. Although these runs were revealing and facilitated process understanding, it was not considered practical to discuss them in the report. Similarly, segmented data (by season or MJO phase) was analysed for all runs, but this analysis was discussed in the body of the report only when it contributed materially to a particular conclusion.

		results, AS, all													results, AS, NEM												
		MLD, m			SST, °C			DCM, °C					MLD, m			SST, °C			DCM, °C				
run	specifics	mean	stdev	RMSE	mean	stdev	RMSE	mean	stdev	RMSE	nRMSE	R	mean	stdev	RMSE	mean	stdev	RMSE	mean	stdev	RMSE	nRMSE	R
OBS	ΔT > 0.2 K	39.3	26.4		27.2	1.55		0.50	0.47				62.8	27.5		26.2	0.94		0.31	0.20			
OBS_dsig	ΔΣ > 0.125 kg m^-3	46.4	27.5										70.1	27.7									
OBS_dd01	Δρ > 0.1 kg m^-3	20.1	8.3										25.9	8.3									
OBS_dT05	ΔT > 0.5 K	48.3	28.1										73.0	28.0									
UnRe	ΔT > 0.2 K	32.6	20.6	18.5	29.9	2.49	3.19	0.77	0.42	0.45	0.91	0.68	43.7	5.8	30.5	27.4	0.64	1.28	0.57	0.22	0.30	0.96	0.70
CTL	at season transitions	28.3	15.6	21.4	28.9	2.06	2.41	0.51	0.36	0.24	0.49	0.87	38.6	7.5	33.7	27.3	0.64	1.23	0.35	0.18	0.10	0.32	0.88
REa	at STs; early start	29.8	19.0	22.1	28.4	1.63	1.64	0.51	0.37	0.27	0.54	0.83	38.6	7.5	33.7	27.3	0.64	1.23	0.35	0.18	0.10	0.32	0.88
dT05	ΔT > 0.5 K	30.4	17.5	26.7	29.1	2.18	2.48	0.77	0.42	0.44	0.89	0.70	43.7	5.8	30.5	27.4	0.64	1.28	0.57	0.22	0.30	0.96	0.70
CTL_dd	Δρ > 0.1 kg m^-3	15.3	5.7	7.6	28.4	1.65	1.77	0.55	0.37	0.24	0.49	0.87	18.8	3.2	10.9	27.3	0.61	1.22	0.38	0.19	0.13	0.40	0.85
dsig	ΔΣ > 0.125 kg m^-3	31.8	18.7	25.8	28.3	1.53	1.46	0.67	0.39	0.36	0.72	0.75	42.4	5.9	31.5	27.3	0.64	1.26	0.47	0.20	0.20	0.63	0.82
24Q	heat fluxes	32.1	20.1	19.9	29.0	2.46	2.60	0.14	0.12	0.58	1.17	0.30	46.3	5.1	29.2	27.2	0.65	1.14	0.06	0.04	0.31	1.00	0.36
24UV	winds	30.2	21.4	20.8	28.7	1.98	1.90	0.48	0.34	0.35	0.71	0.68	39.2	7.5	33.2	27.3	0.64	1.21	0.32	0.12	0.13	0.41	0.80
24QUV	all	30.2	20.1	23.7	28.5	2.71	3.10	0.15	0.19	0.60	1.20	0.18	47.6	9.6	25.3	26.7	0.95	0.65	0.07	0.04	0.31	0.99	0.25
24Q_dd	heat fluxes	17.5	7.3	6.8	28.7	2.02	2.11	0.15	0.17	0.59	1.19	0.22	24.1	2.7	8.1	27.2	0.64	1.15	0.06	0.04	0.32	1.01	0.28
24UV_dd	winds	15.4	5.8	7.5	28.4	1.68	1.74	0.52	0.34	0.31	0.62	0.77	19.1	3.1	10.8	27.3	0.60	1.17	0.35	0.14	0.13	0.41	0.79
24QUV_dd	all	17.2	7.6	6.9	29.3	2.63	2.99	0.14	0.13	0.59	1.18	0.26	24.1	2.8	8.1	27.2	0.64	1.15	0.06	0.04	0.31	1.00	0.32
12HR	all; centred on 0000,120	29.6	16.9	19.3	28.4	1.62	1.62	0.40	0.28	0.34	0.68	0.76	41.2	8.7	31.5	27.3	0.64	1.21	0.23	0.14	0.17	0.53	0.70
12HR_off	all; centred on 0600,180	30.3	19.5	20.5	28.6	2.01	1.78	0.17	0.20	0.57	1.15	0.25	45.5	7.3	28.9	27.2	0.64	1.15	0.07	0.06	0.30	0.96	0.58
24HR	all; centred on 1200	32.3	23.2	21.2	28.7	2.09	1.98	0.14	0.17	0.60	1.21	0.14	46.4	6.1	28.8	27.2	0.65	1.13	0.06	0.03	0.32	1.02	0.27
12HR_dd	all; centred on 0000,120	15.3	5.9	7.7	28.4	1.66	1.58	0.44	0.32	0.31	0.63	0.78	19.6	4.0	11.0	27.3	0.64	1.22	0.25	0.17	0.15	0.49	0.74
12HR_off_dd	all; centred on 0600,180	17.1	7.3	6.9	29.0	2.28	2.42	0.17	0.16	0.55	1.11	0.41	23.6	3.3	8.3	27.2	0.64	1.15	0.07	0.06	0.30	0.96	0.60
24HR_dd	all; centred on 1200	17.2	7.5	6.9	29.1	2.35	2.54	0.14	0.13	0.59	1.19	0.23	24.1	2.7	8.1	27.2	0.63	1.12	0.06	0.04	0.32	1.02	0.23
noFW_dT	no freshwater flux	31.1	21.7	21.4	28.4	1.63	1.59	0.51	0.37	0.27	0.55	0.82	38.3	8.5	33.6	27.3	0.63	1.23	0.35	0.17	0.10	0.32	0.88
noFW_dd	no freshwater flux	15.4	5.7	33.6	28.4	1.74	1.85	0.54	0.38	0.25	0.50	0.86	18.9	3.4	51.2	27.2	0.59	1.17	0.39	0.19	0.13	0.41	0.85
halfUV_dT	wind forcing halved	22.4	14.9	23.8	29.3	2.45	2.75	0.63	0.44	0.26	0.53	0.88	36.0	8.5	35.5	27.3	0.63	1.27	0.43	0.21	0.16	0.52	0.86
2xUV_dT	wind forcing doubled	42.6	32.2	28.4	27.2	1.19	0.93	0.43	0.32	0.36	0.73	0.67	40.6	7.6	31.7	27.2	0.66	1.15	0.29	0.11	0.13	0.41	0.80
CC13	13m	14.5	3.5	37.2	25.8	5.37	5.27	0.21	0.04	0.54	1.10	0.38	15.6	6.4	55.3	23.3	3.58	3.96	0.20	0.04	0.23	0.75	-0.03
CC27	27m	28.8	3.2	29.3	30.1	4.31	4.92	0.12	0.03	0.60	1.21	0.47	30.6	5.5	44.0	26.4	1.32	0.59	0.11	0.02	0.29	0.91	0.15
CC40	40m	41.2	0.6	26.9	30.4	3.52	4.59	0.09	0.02	0.62	1.26	0.50	41.6	1.0	35.1	27.1	0.76	1.04	0.07	0.01	0.31	0.99	0.25
CC54/55	54m or 55m	55.0	0.0	30.7	29.8	2.87	3.74	0.07	0.01	0.64	1.29	0.50	55.0	0.0	28.4	27.0	0.51	0.99	0.06	0.01	0.32	1.03	0.30
CG	convection, gradient	32.3	23.2	21.2	28.7	2.09	1.98	0.14	0.17	0.60	1.21	0.14	46.4	6.1	28.8	27.2	0.65	1.13	0.06	0.03	0.32	1.02	0.27
BG	bulk, gradient	21.8	14.4	31.7	28.4	1.62	1.71	0.61	0.35	0.26	0.53	0.88	16.2	5.2	54.6	26.9	0.66	0.77	0.45	0.18	0.18	0.56	0.84
BC	bulk, convection	28.1	15.4	20.5	30.2	2.69	3.70	0.52	0.37	0.24	0.49	0.87	37.6	8.0	34.2	27.3	0.63	1.24	0.35	0.18	0.10	0.32	0.88
B	bulk	22.8	18.0	17.5	27.8	1.53	1.22	0.61	0.37	0.30	0.60	0.82	15.9	4.6	13.0	26.8	0.77	0.74	0.44	0.18	0.17	0.55	0.81
C	convection	14.0	15.2	14.6	38.0	16.7	19.7	1.21	0.53	0.89	1.79	0.42	30.0	10.2	13.8	27.4	0.65	1.36	0.70	0.33	0.47	1.50	0.55
G	gradient	8.4	5.9	14.0	29.8	3.73	4.06	0.76	0.47	0.35	0.71	0.87	7.7	3.8	19.9	26.4	1.26	0.56	0.56	0.23	0.28	0.91	0.80

		results, AS, IM											results, AS, SWM										
		MLD, m			SST, °C			DCM, °C					MLD, m			SST, °C			DCM, °C				
run	specifics	mean	stdev	RMSE	mean	stdev	RMSE	mean	stdev	RMSE	nRMSE	R	mean	stdev	nRMSE	mean	stdev	RMSE	mean	stdev	RMSE	nRMSE	R
OBS	ΔT > 0.2 K	20.9	14.9		28.1	1.84		0.83	0.64				40.1	20.1		27.2	1.17		0.33	0.23			
OBS_dsig	ΔΣ > 0.125 kg m^-3	32.2	23.4										43.4	18.5									
OBS_dd01	Δρ > 0.1 kg m^-3	15.1	7.8										20.0	5.5									
OBS_dT05	ΔT > 0.5 K	33.7	23.9										45.0	18.5									
UnRe	ΔT > 0.2 K	16.6	11.7	9.6	30.6	2.32	2.57	1.05	0.47	0.50	0.60	0.69	39.1	25.2	10.2	31.3	2.02	4.53	0.68	0.36	0.51	1.53	0.27
CTL	at season transitions	16.8	14.2	8.0	28.3	1.86	0.36	0.78	0.45	0.35	0.42	0.86	31.0	14.7	16.6	30.8	1.49	3.86	0.38	0.18	0.19	0.59	0.59
REa	at STs; early start	16.5	14.0	8.0	28.2	1.86	0.37	0.79	0.45	0.35	0.42	0.85	35.3	22.6	19.1	29.5	1.26	2.48	0.39	0.24	0.27	0.82	0.37
dT05	ΔT > 0.5 K	20.2	19.2	11.6	28.9	2.28	1.16	1.05	0.50	0.54	0.64	0.64	29.4	14.9	18.0	30.7	1.71	3.80	0.66	0.29	0.44	1.32	0.40
CTL_dd	Δρ > 0.1 kg m^-3	10.6	5.4	6.2	28.2	1.75	0.35	0.84	0.44	0.33	0.39	0.88	17.0	4.6	4.9	29.6	1.37	2.71	0.40	0.21	0.21	0.65	0.58
dsig	ΔΣ > 0.125 kg m^-3	19.1	16.9	7.4	28.3	1.83	0.49	0.95	0.43	0.41	0.49	0.78	35.2	20.4	20.1	29.2	1.23	2.10	0.56	0.33	0.40	1.21	0.35
24Q	heat fluxes	21.5	23.9	16.0	28.8	2.70	1.21	0.16	0.15	0.91	1.09	0.30	30.8	16.8	12.5	30.8	1.93	4.05	0.19	0.12	0.26	0.79	0.33
24UV	winds	15.7	14.5	7.9	28.8	2.38	0.92	0.73	0.42	0.53	0.63	0.59	36.8	26.8	15.4	29.7	1.59	2.82	0.39	0.24	0.26	0.80	0.38
24QUV	all	29.6	24.4	16.6	26.8	0.89	1.88	0.20	0.29	0.94	1.13	0.04	16.6	6.5	27.8	31.5	2.13	4.80	0.17	0.11	0.26	0.78	0.44
24Q_dd	heat fluxes	12.9	6.8	6.2	28.3	2.13	0.59	0.17	0.17	0.92	1.10	0.20	16.5	6.6	6.1	30.2	1.54	3.31	0.19	0.20	0.29	0.89	0.28
24UV_dd	winds	10.7	5.7	6.1	28.2	1.87	0.29	0.79	0.39	0.45	0.54	0.72	16.9	4.6	5.0	29.5	1.35	2.68	0.41	0.23	0.25	0.75	0.46
24QUV_dd	all	11.9	6.9	6.6	28.7	2.43	0.93	0.17	0.17	0.92	1.10	0.15	16.6	6.5	6.2	31.5	2.13	4.80	0.17	0.11	0.26	0.78	0.44
12HR	all; centred on 0000,120	16.3	12.6	7.2	28.1	1.80	0.31	0.61	0.32	0.50	0.60	0.77	32.8	17.1	13.0	29.5	1.28	2.45	0.34	0.20	0.24	0.72	0.39
12HR_off	all; centred on 0600,180	18.3	21.5	14.8	29.0	2.76	1.38	0.22	0.14	0.87	1.04	0.26	29.6	15.8	16.7	29.5	1.11	2.43	0.21	0.27	0.34	1.04	0.18
24HR	all; centred on 1200	20.0	23.5	14.6	28.6	2.36	0.83	0.16	0.16	0.92	1.11	0.16	32.8	25.2	19.0	29.9	1.81	3.04	0.19	0.21	0.33	0.99	0.10
12HR_dd	all; centred on 0000,120	10.5	5.2	6.1	28.4	2.06	0.55	0.68	0.34	0.44	0.52	0.84	16.3	4.5	5.3	29.4	1.15	2.34	0.36	0.24	0.27	0.80	0.34
12HR_off_dd	all; centred on 0600,180	12.1	6.4	6.2	28.8	2.56	1.11	0.23	0.15	0.85	1.02	0.39	16.7	6.4	6.1	30.6	1.64	3.75	0.19	0.18	0.27	0.83	0.32
24HR_dd	all; centred on 1200	11.9	6.8	6.7	28.8	2.44	0.98	0.16	0.16	0.93	1.11	0.13	16.6	6.4	6.0	30.8	1.87	4.00	0.17	0.12	0.25	0.77	0.47
noFW_dT	no freshwater flux	16.1	13.3	7.3	28.3	1.96	0.46	0.78	0.44	0.36	0.43	0.84	39.4	27.5	17.0	29.3	1.25	2.37	0.38	0.26	0.27	0.83	0.40
noFW_dd	no freshwater flux	10.8	5.4	13.7	28.1	1.73	0.37	0.84	0.44	0.34	0.41	0.86	17.1	4.6	28.3	29.7	1.59	2.88	0.37	0.21	0.21	0.65	0.55
halfUV_dT	wind forcing halved	11.1	11.0	11.0	29.0	2.50	1.22	0.99	0.48	0.31	0.37	0.91	22.2	13.2	20.5	31.1	2.01	4.28	0.46	0.31	0.28	0.84	0.63
2xUV_dT	wind forcing doubled	22.6	18.1	11.7	27.6	1.50	0.79	0.62	0.40	0.51	0.61	0.71	63.2	41.4	35.8	26.9	1.10	0.85	0.36	0.28	0.33	0.99	0.18
CC13	13m	14.0	0.0	16.3	22.4	2.93	5.84	0.23	0.03	0.87	1.04	0.52	14.0	0.0	32.9	31.1	4.21	5.62	0.20	0.05	0.25	0.75	0.40
CC27	27m	28.0	0.0	16.4	28.2	2.26	0.75	0.13	0.02	0.94	1.12	0.55	28.1	0.2	23.4	34.9	2.72	8.16	0.12	0.03	0.30	0.92	0.39
CC40	40m	41.2	0.5	25.1	29.0	1.80	1.08	0.10	0.01	0.97	1.16	0.54	41.0	0.0	20.0	34.4	2.05	7.54	0.08	0.02	0.33	0.99	0.39
CC54/55	54m or 55m	55.0	0.0	37.2	28.7	1.45	0.87	0.08	0.01	0.98	1.18	0.54	55.0	0.0	24.9	33.0	1.61	6.12	0.07	0.02	0.34	1.04	0.38
CG	convection, gradient	20.0	23.5	14.6	28.6	2.36	0.83	0.16	0.16	0.92	1.11	0.16	32.8	25.2	19.0	29.9	1.81	3.04	0.19	0.21	0.33	0.99	0.10
BG	bulk, gradient	12.5	6.0	15.0	28.7	1.58	0.75	0.89	0.42	0.34	0.41	0.88	35.3	15.2	11.5	29.4	1.27	2.67	0.47	0.16	0.24	0.72	0.53
BC	bulk, convection	15.0	10.1	9.5	30.0	2.03	1.96	0.79	0.45	0.32	0.38	0.89	33.0	15.8	11.3	32.7	1.66	5.76	0.39	0.22	0.24	0.73	0.46
B	bulk	11.9	6.5	6.9	28.0	1.86	0.46	0.90	0.44	0.37	0.44	0.83	38.8	20.8	25.8	28.5	1.23	1.87	0.49	0.23	0.31	0.93	0.34
C	convection	6.0	10.2	13.7	30.6	3.84	3.28	1.45	0.27	0.80	0.95	0.58	8.5	12.6	16.0	53.8	19.2	32.7	1.39	0.56	1.18	3.59	0.40
G	gradient	4.8	3.2	12.0	29.2	2.84	1.57	1.17	0.52	0.45	0.53	0.87	12.4	6.9	9.1	33.0	3.19	6.56	0.55	0.27	0.30	0.91	0.65

Figure 15: Full statistical data, AS. "UV" is used rather than τ in run names that modified wind forcing. For mean and σ, darker colours indicate larger values. For accuracy measures (R, RMSE, nRMSE), green is more accurate and red is less accurate. MLD nRMSE was calculated separately. Direct visual comparison between columns/sections may not always be possible due to selective formatting. Some outliers have been excluded from conditional formatting.

Return to 4.2.3: Statistical measures

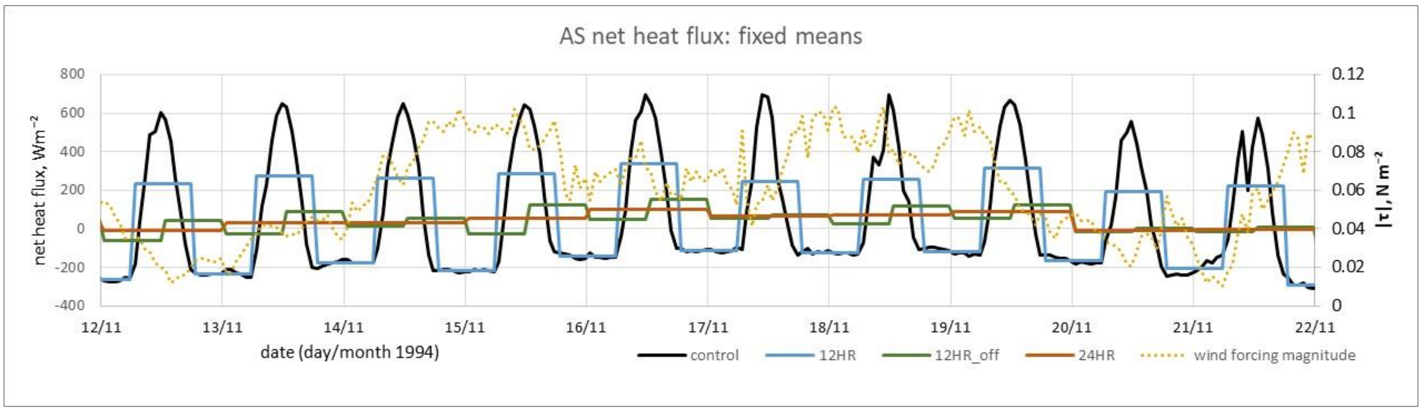
7.4.5 Full model statistics for all runs, COARE dataset

		results, COARE, all											results, COARE, warming											results, COARE, cooling											results, COARE, neutral										
		MLD, m			SST, °C			DCM, °C					MLD, m			SST, °C			DCM, °C					MLD, m			SST, °C			DCM, °C					MLD, m			SST, °C			DCM, °C				
run	specifics	mean	stdev	RMSE	mean	stdev	RMSE	mean	stdev	RMSE	nRMSE	R	mean	stdev	RMSE	mean	stdev	RMSE	mean	stdev	RMSE	nRMSE	R	mean	stdev	RMSE	mean	stdev	RMSE	mean	stdev	RMSE	nRMSE	R	mean	stdev	RMSE	mean	stdev	RMSE	mean	stdev	RMSE	nRMSE	R
OBS	ΔT > 0.2 K	38.8	22.0		29.4	0.40		0.62	0.52				15.2	12.7		29.6	0.47		1.15	0.58				54.1	23.1		29.3	0.39		0.33	0.24				43.1	19.6		29.3	0.31		0.51	0.43			
OBS_dsig	ΔΣ > 0.125 kg m^-3	41.8	21.2										18.9	13.3										49.4	19.1										48.7	17.5									
OBS_dd	Δρ > 0.1 kg m^-3	14.8	5.8										8.3	4.1										17.5	5.0										16.6	4.8									
OBS_dT05	ΔT > 0.5 K	52.9	22.4										32.3	20.3										58.4	20.0										59.5	18.6									
CTL	ΔT > 0.2 K	27.6	11.1	18.1	29.9	0.34	0.53	0.59	0.31	0.32	0.52	0.84	16.2	10.2	8.2	30.0	0.38	0.47	0.88	0.20	0.52	0.45	0.64	31.0	8.1	23.2	29.9	0.31	0.66	0.45	0.29	0.18	0.55	0.69	31.3	9.6	17.9	29.8	3.54	0.49	0.51	0.28	0.23	0.44	0.87
dT05	ΔT > 0.5 K	34.7	10.4	23.8	29.9	0.35	0.57	0.75	0.29	0.39	0.63	0.73	24.1	9.9	15.3	30.1	0.39	0.53	0.95	0.17	0.51	0.44	0.58	36.3	7.4	27.1	29.9	0.31	0.66	0.63	0.29	0.32	0.99	0.48	38.7	9.5	25.1	29.8	3.55	0.53	0.72	0.29	0.34	0.66	0.77
CTL_dd	Δρ > 0.1 kg m^-3	15.4	5.8	3.1	28.6	0.80	1.08	0.67	0.37	0.39	0.64	0.67	9.0	4.9	3.9	28.9	0.37	0.78	0.90	0.19	0.51	0.44	0.66	17.8	5.0	3.0	28.8	0.31	0.54	0.51	0.31	0.21	0.64	0.74	17.1	4.8	2.7	28.4	3.51	1.34	0.65	0.41	0.38	0.74	0.64
dsig	ΔΣ > 0.125 kg m^-3	30.1	11.4	18.2	30.0	0.34	0.71	0.68	0.31	0.35	0.57	0.72	18.6	10.3	10.5	30.2	0.37	0.64	0.93	0.18	0.51	0.45	0.58	30.6	10.2	22.6	30.1	0.32	0.81	0.54	0.29	0.25	0.78	0.56	35.2	10.6	18.3	30.0	6.08	0.69	0.63	0.31	0.27	0.53	0.35
24Q_dT	heat fluxes	42.5	15.9	18.0	30.0	0.27	0.73	0.14	0.10	0.69	1.11	0.44	27.2	17.5	19.2	30.1	0.30	0.54	0.17	0.08	1.10	0.96	0.12	44.5	11.6	10.9	30.1	0.17	0.84	0.13	0.07	0.27	0.82	0.53	48.6	14.1	19.0	30.0	5.01	0.73	0.14	0.12	0.52	1.02	0.54
24UV_dT	winds	28.0	11.8	17.2	29.8	0.31	0.50	0.52	0.28	0.40	0.64	0.72	15.4	9.7	8.4	30.0	0.35	0.43	0.81	0.22	0.57	0.50	0.55	31.9	10.5	20.6	29.9	0.29	0.62	0.38	0.22	0.19	0.60	0.50	31.8	9.6	17.7	29.7	3.54	0.46	0.45	0.24	0.35	0.68	0.61
24QUV_dT	all	47.0	23.6	24.4	29.2	0.50	0.40	0.15	0.13	0.71	1.16	0.09	28.8	18.3	21.8	29.5	0.51	0.31	0.16	0.08	1.10	0.96	0.13	48.7	21.2	11.1	29.4	0.65	0.36	0.17	0.21	0.32	0.98	0.11	54.5	23.2	28.5	29.1	3.46	0.44	0.13	0.09	0.57	1.12	0.11
24QUVb_dT	all; late start	35.6	12.7	19.6	30.1	0.28	0.78	0.14	0.09	0.69	1.12	0.45	26.7	14.5	17.9	30.1	0.29	0.56	0.16	0.08	1.10	0.96	0.15	38.0	9.0	17.5	30.1	0.18	0.90	0.13	0.06	0.27	0.82	0.62	38.5	12.8	20.2	30.1	5.02	0.79	0.13	0.10	0.53	1.04	0.52
24Q_dd	heat fluxes	19.9	6.2	6.8	28.8	0.73	0.87	0.20	0.21	0.70	1.14	0.01	13.3	6.2	7.8	29.1	0.41	0.59	0.18	0.10	1.10	0.96	-0.01	20.9	5.5	4.8	29.1	0.46	0.25	0.20	0.17	0.24	0.75	0.40	22.3	4.7	7.1	28.6	3.50	1.12	0.22	0.25	0.58	1.13	0.02
24UV_dd	winds	15.4	5.9	3.2	28.5	1.29	1.45	0.63	0.34	0.43	0.70	0.57	9.0	4.8	4.7	29.2	1.20	1.02	0.87	0.22	0.54	0.47	0.53	18.4	4.9	3.0	29.0	1.32	1.03	0.50	0.45	0.46	1.42	0.20	17.0	4.9	2.3	27.9	3.47	1.73	0.58	0.26	0.33	0.65	0.65
24QUV_dd	all	19.3	6.6	6.7	29.1	0.27	0.45	0.19	0.18	0.70	1.14	0.02	12.7	6.7	7.8	29.0	0.28	0.74	0.17	0.10	1.11	0.97	-0.01	20.2	5.5	4.6	29.0	0.22	0.31	0.20	0.17	0.26	0.79	0.31	21.9	5.3	6.9	29.1	3.47	0.32	0.21	0.21	0.55	1.08	0.07
12HR	all; centred on 0000,120	34.1	20.1	17.5	29.6	0.55	0.51	0.42	0.26	0.45	0.72	0.68	14.1	8.6	7.9	30.1	0.40	0.50	0.62	0.15	0.69	0.60	0.72	29.0	12.2	22.6	30.0	0.32	0.77	0.38	0.35	0.32	0.99	0.42	45.3	19.3	17.2	29.2	3.49	0.33	0.35	0.21	0.32	0.63	0.84
12HR_off	all; centred on 0600,180	33.8	11.8	17.4	29.7	0.27	0.41	0.16	0.10	0.68	1.11	0.40	23.0	12.0	12.7	29.8	0.29	0.32	0.22	0.12	1.05	0.92	0.12	33.2	10.3	20.0	29.8	0.23	0.52	0.14	0.08	0.24	0.75	0.76	38.9	9.7	17.3	29.6	3.53	0.39	0.13	0.08	0.57	1.11	0.22
24HR	all; centred on 1200	51.1	36.3	32.2	28.4	1.24	1.48	0.15	0.23	0.74	1.20	0.03	29.7	30.1	31.5	29.0	1.19	1.13	0.16	0.07	1.11	0.97	0.06	59.8	44.4	33.8	28.8	1.38	1.17	0.22	0.45	0.45	1.37	0.29	56.7	31.1	30.4	27.9	3.46	1.71	0.12	0.08	0.59	1.15	-0.03
24HR	all; centred on 1200	32.5	11.2	19.5	29.7	0.26	0.42	0.14	0.08	0.71	1.15	0.23	22.8	12.6	15.8	29.8	0.30	0.30	0.16	0.07	1.11	0.97	-0.02	33.4	8.8	21.0	29.8	0.24	0.56	0.14	0.08	0.24	0.74	0.80	36.4	9.7	19.5	29.6	3.52	0.40	0.12	0.08	0.58	1.13	0.14
12HR_dd	all; centred on 0000,120	15.9	6.2	3.3	28.7	0.80	0.98	0.52	0.31	0.45	0.73	0.56	9.0	4.5	4.2	29.1	0.31	0.57	0.69	0.17	0.65	0.57	0.53	18.8	4.9	3.3	29.0	0.24	0.37	0.37	0.25	0.17	0.52	0.73	17.6	5.3	2.9	28.4	3.51	1.26	0.50	0.34	0.40	0.78	0.48
12HR_off_dd	all; centred on 0600,180	19.4	6.3	6.3	29.7	0.27	0.38	0.20	0.17	0.68	1.10	0.15	12.6	6.5	7.0	29.8	0.30	0.28	0.23	0.12	1.05	0.92	0.03	20.7	4.1	4.6	29.8	0.25	0.52	0.19	0.17	0.23	0.70	0.55	21.9	5.1	6.6	29.6	3.52	0.33	0.19	0.19	0.56	1.09	0.08
24HR_dd	all; centred on 1200	20.1	6.2	6.8	29.6	0.26	0.34	0.19	0.17	0.69	1.13	0.09	12.8	6.5	7.2	29.7	0.31	0.27	0.17	0.09	1.11	0.97	0.01	20.9	4.6	4.3	29.7	0.24	0.51	0.21	0.17	0.22	0.68	0.55	23.0	4.3	7.5	29.5	3.50	0.26	0.20	0.20	0.55	1.07	0.13
noFW_dT	no freshwater flux	28.0	11.9	16.9	29.9	0.33	0.52	0.56	0.31	0.33	0.53	0.83	15.8	10.4	7.8	30.0	0.37	0.47	0.87	0.20	0.52	0.46	0.66	31.1	9.8	21.6	29.9	0.33	0.66	0.42	0.27	0.19	0.60	0.59	32.1	10.0	16.8	29.8	3.54	0.47	0.48	0.28	0.24	0.46	0.86
noFW_dd	no freshwater flux	16.3	5.8	28.8	28.6	0.51	0.94	0.59	0.32	0.33	0.54	0.81	10.2	5.0	10.9	28.6	0.46	1.15	0.87	0.19	0.52	0.46	0.67	18.8	4.7	34.6	28.3	0.29	0.97	0.47	0.29	0.20	0.63	0.62	17.9	5.0	30.2	28.8	3.46	0.81	0.52	0.31	0.25	0.48	0.82
halfUV_dT	wind forcing halved	21.4	9.9	22.6	30.0	0.37	0.66	0.68	0.33	0.33	0.54	0.81	10.9	7.9	8.6	30.2	0.39	0.61	1.00	0.18	0.50	0.43	0.54	24.0	9.5	27.8	30.0	0.32	0.73	0.54	0.30	0.23	0.71	0.66	24.9	7.8	23.4	29.9	3.57	0.65	0.61	0.31	0.25	0.49	0.84
2xUV_dT	wind forcing doubled	50.8	33.0	26.6	27.6	0.84	2.00	0.54	0.36	0.51	0.83	0.55	34.2	28.1	23.0	27.3	12.4	2.05	0.68	0.36	0.54	0.47	0.22	58.5	38.6	23.6	27.3	10.8	1.87	0.41	0.26	0.16	0.49	0.46	59.0	31.5	12.5	28.4	#####	0.68	0.56	0.38	0.31	0.60	0.31
CC13	13m	14.4	2.1	33.3	27.7	1.27	2.09	0.17	0.05	0.71	1.16	-0.39	14.0	0.0	12.5	28.2	0.82	1.55	0.13	0.02	1.14	1.00	-0.14	14.0	0.0	40.1	28.2	0.73	1.14	0.19	0.06	0.27	0.84	-0.45	14.7	3.4	34.8	27.2	3.52	2.56	0.18	0.05	0.55	1.08	-0.11
CC27	27m	28.2	0.8	24.8	29.6	0.25	0.37	0.08	0.02	0.76	1.23	-0.34	28.0	0.0	17.7	29.6	0.26	0.29	0.07	0.01	1.19	1.04	-0.11	28.0	0.0	28.2	29.7	0.27	0.51	0.09	0.03	0.31	0.96	-0.45	28.3	3.5	24.7	29.5	3.51	0.33	0.09	0.03	0.61	1.18	-0.10
CC39	40m	40.4	0.7	22.1	29.8	0.23	0.56	0.06	0.02	0.78	1.26	-0.31	40.1	0.3	27.5	29.7	0.20	0.38	0.05	0.01	1.21	1.05	-0.08	40.1	0.3	19.3	29.9	0.18	0.65	0.06	0.02	0.33	1.01	-0.46	40.6	4.9	19.6	29.8	3.55	0.58	0.06	0.02	0.62	1.22	-0.09
CC54/55	54m or 55m	56.0	0.0	28.1	29.4	0.23	0.41	0.04	0.01	0.79	1.28	-0.29	56.0	0.0	42.0	29.3	0.17	0.54	0.04	0.00	1.22	1.06	-0.05	56.0	0.0	14.4	29.4	0.13	0.35	0.05	0.01	0.34	1.05	-0.46	56.0	6.6	23.5	29.4	3.50	0.37	0.04	0.01	0.64	1.24	-0.09
CG	convection, gradient	23.3	10.5	20.9	29.9	0.36	0.59	0.68	0.32	0.33	0.53	0.82	12.5	8.3	7.7	30.1	0.39	0.55	0.97	0.17	0.49	0.43	0.64	26.5	9.7	25.7	29.9	0.33	0.68	0.53	0.30	0.23	0.70	0.67	26.6	8.9	21.6	29.8	3.56	0.56	0.61	0.30	0.25	0.48	0.86
BG	bulk, gradient	29.6	13.6	21.4	27.5	0.60	1.95	0.68	0.31	0.36	0.58	0.76	24.7	14.5	16.8	27.6	0.36	2.03	0.94	0.19	0.50	0.44	0.61	31.5	13.8	22.2	27.4	0.42	1.89	0.58	0.29	0.25	0.78	0.65	30.9	12.9	22.0	27.5	3.34	1.93	0.61	0.30	0.30	0.59	0.74
BC	bulk, convection	29.4	13.6	15.3	29.8	0.35	0.50	0.59	0.31	0.32	0.52	0.84	15.9	10.8	8.4	30.0	0.39	0.46	0.87	0.20	0.52	0.45	0.66	35.0	12.6	17.3	29.8	0.38	0.58	0.47	0.30	0.20	0.60	0.66	32.9	11.5	16.1	29.8	3.54	0.48	0.52	0.28	0.22	0.44	0.88
B	bulk	27.2	12.5	22.6	27.9	0.82	1.65	0.69	0.33	0.37	0.60	0.72	21.3	13.5	15.2	28.3	0.67	1.42	0.93	0.19	0.49	0.43	0.64	31.2	13.1	22.7	27.9	0.86	1.48	0.60	0.37	0.36	1.12	0.40	28.1	11.0	24.2	27.7	3.38	1.80	0.61	0.32	0.28	0.55	0.79
C	convection	13.2	8.9	29.8	30.0	0.46	0.70	0.95	0.48	0.64	1.04	0.41	6.5	6.5	12.3	30.3	0.42	0.72	1.11	0.16	0.53	0.46	0.25	15.4	8.5	36.0	30.0	0.33	0.75	0.76	0.35	0.45	1.39	0.55	15.3	8.6	30.8	29.8	3.57	0.66	0.97	0.59	0.72	1.40	0.43
G	gradient	9.2	3.9	35.9	29.7	0.34	0.44	0.82	0.36	0.40	0.64	0.75	6.7	3.9	14.4	29.8	0.41	0.34	1.16	0.24	0.48	0.42	0.47	10.6	3.4	42.2	29.8	0.33	0.59	0.64	0.28	0.30	0.93	0.63	9.8	3.7	37.9	29.7	3.53	0.39	0.76	0.35	0.38	0.73	0.73

Figure 16: As previous figure.

Return to 4.2.3: Statistical measures

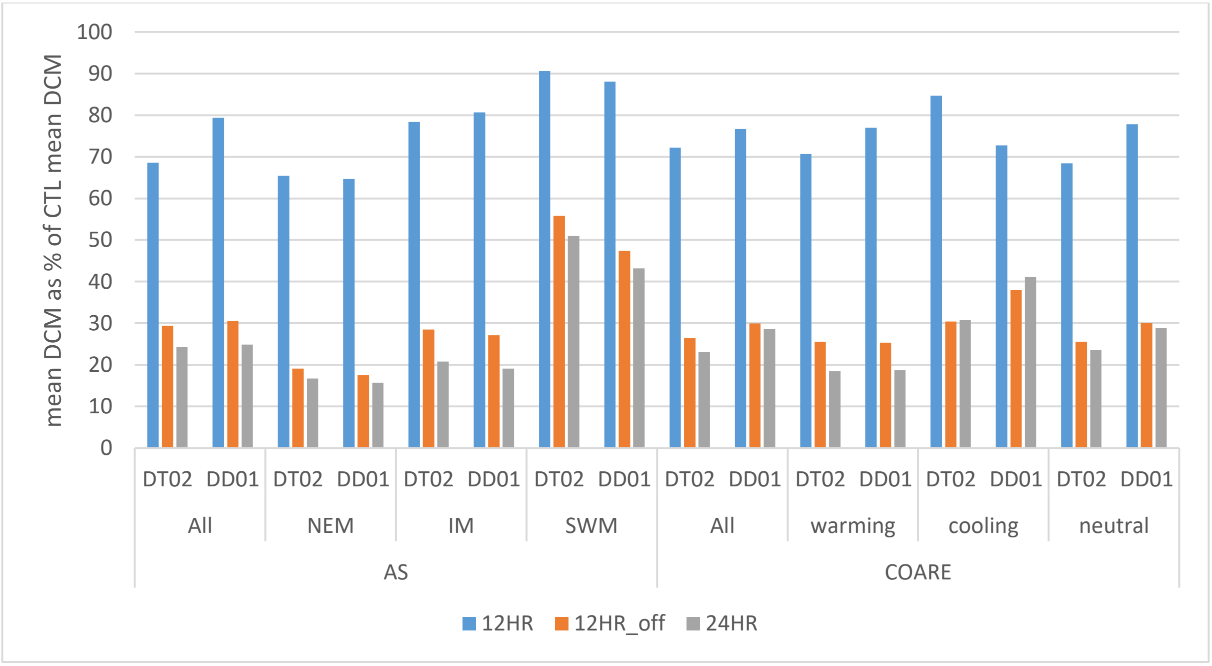
7.5 AS fixed mean heat fluxes



The impact of the various fixed mean periods is functionally identical in AS and COARE. Note again the lack of diurnality in τ.

Return to 5.2.1: Fixed means

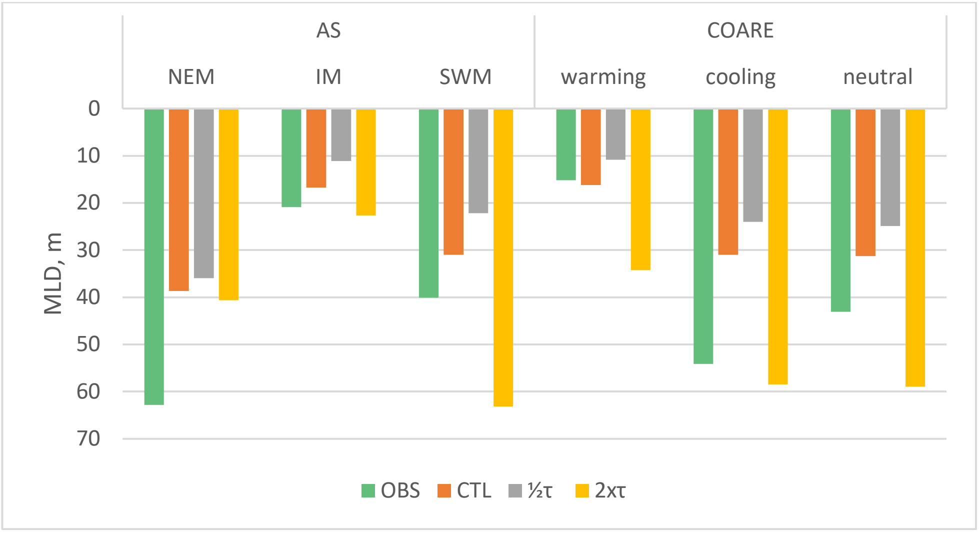
7.6 Fixed means: segmented data



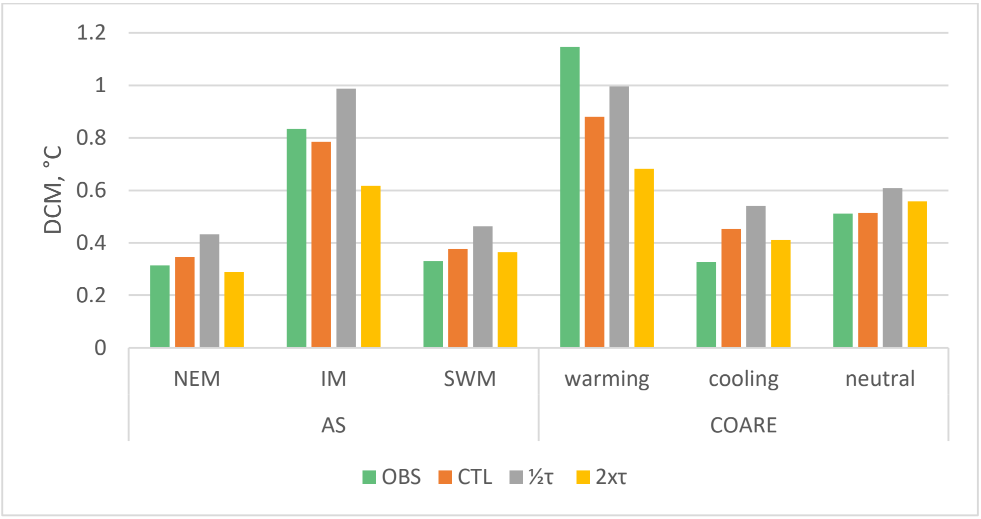
Mean DCM as percentage of mean control DCM when model is forced with fixed mean forcings for each period type, dataset and MLD criterion. Note consistency of pattern across all seasons/phases, with the exception of a slight, unexplained reversal during cooling phases in COARE.

Return to 5.2.1: Fixed means

7.7 Impact of momentum flux alteration on MLD and DCM: segmented data



In all seasons (AS) and phases (COARE), MLD shoals in ½τ and deepens in 2xτ in comparison to CTL.



In all seasons (AS) and phases (COARE) for which full data is available, DCM increases in ½τ and decreases in 2xτ in comparison to CTL. The apparent exception is during neutral phases in COARE, but this may be because it was only possible to fully model the first neutral phase: the ML mixed off the bottom of the column towards the end of the second cooling phase, and the model did not recover.

Return to 5.3.2: Momentum flux alteration

7.8 Model code

7.8.1 Main

This is the front end code that steps through the model iterations

```
from os.path import join
from column import Column                  # custom class, defined below
from forcings import Forcings              # custom class, not included here
import gsw
from loader_rdx import plotcolumn          # plotter function, not included here
from day_log import Day_log                # custom logger class, not included here
from datetime import datetime
from parameters import as_source_file_dict as source_file_dict          # dataset parameters (lat, long, etc.)
import nc_TG                               # custom function to write profiles out to netCDF. Not included here.

as_forcings_dir = r"D:\UKuni\3rdYr\MEP\cw\data\ArabianSea\forcings"        # input forcings folder
as_forcings_filename = "AS_forcings_94_10_19_1800on_EPIC_noFW.csv"       # input forcings filename
as_forcings_csv = join(as_forcings_dir, as_forcings_filename)
```

```python
coare_forcings_dir = r"D:\UKuni\3rdYr\MEP\cw\data\COARE\forcings"
coare_forcings_filename = "COARE_forcings_EPIC_i1_noFW.csv"
coare_forcings_csv = join(coare_forcings_dir, coare_forcings_filename)

def run_pwp(obs_profile_source, obs_format, forcings_csv, depthlimit, interval=None, plotstep=None, dates=None,
            recordingstyle="rate", wind_as_vel=False, ml_method="dd_by_dz", m_show="all", mode="run",
            show_initial=False, show_obs=False, nctag="", cc_depth=None):

    # step 0) read in profile data, set mode, read in forcing data
    # construct column with initial T, S, U, V profiles taken from netcdf
    col = Column(obs_profile_source, obs_format=obs_format, dates=dates)
    # density is calculated as part of the column object initialisation, and additional properties dt, dz and latitude
    # are set from the param dict

    col.mode = mode # set column mode ("run" or "test")

    # read in forcing data. wind_as_vels=False if wind data are stored as stresses.
    forcings = Forcings(forcings_csv, wind_as_vel=wind_as_vel, dates=dates, recordingstyle=recordingstyle)
    col.n_timesteps = len(forcings)
    if plotstep == None or plotstep == 0:
        plotstep = col.n_timesteps

    day_log = Day_log() # construct a day_log object to record average day SST and MLD

    for n in range(col.n_timesteps):
        set_of_forcings = forcings.getset(n) # get set of forcings for this timestep

        # step 1) apply heat fluxes
        col.apply_sltq(set_of_forcings["sltq"])     # apply sensible, latent and terrestrial heat fluxes
        col.apply_solq(set_of_forcings["solq"])     # apply solar heat flux

        # step 2) apply freshwater flux
        col.sprofile.apply_freshwater(set_of_forcings["fw"]) # apply freshwater forcing

        # step 3) recalculate density profile; run convective mixing
        col.calculate_dprofile_gsw(loop=n)
        if ml_method == "dsigma_from_surface": # make potential density profile if needed
            col.calculate_sigmaprofile()
        col.mix_convective(m_show, cc_depth=cc_depth)

        # step 4) rotate; apply momentum flux to mixed layer
        # first, find and log mixed layer base index
        col.set_ml_base_idx(ml_method = ml_method) # this is now accessible as col.ml_base_idx

        coriolis_param = gsw.f(col.lat)                                  # get rotation parameter
        half_rotation = -coriolis_param * col.dt / 2
        col.rotate_by_angle(half_rotation)                                 # do half of rotation
        col.apply_wind_stresses([set_of_forcings["u"], set_of_forcings["v"]])   # apply wind stress
        col.rotate_by_angle(half_rotation)                                 # do second half of rotation

        # step 5) bulk Richardson mixing
        col.set_bulk_richardson()                                        # calculate bulk richardson number
        col.bulk_r_mix(m_show, day_log, nudgepoint=28, nudgeval=0.5)          # check for and remove any instabilities
        # bulk mixing occasionally goes too far and mixes in e.g. 90 column levels in a row in a way that disagrees
        # with observations. These nudge values allow one to avoid that. See column.py for details.

        # step 6) gradient richardson mixing
        col.set_grprofile()                          # calculate gradient richardson number for every level in column
        col.gr_mix_col(m_show, day_log)                # if any are subcritical, mix till they aren't

        # step 7) readouts and plotting
        pc = round(100 * (n + 1) / col.n_timesteps, 1)                 # get completion %
        print(f"loop {n} ({pc}% complete). ML thickness = {int(col.ml_thickness)}m; bulk Richardson no. = "
              f"{round(col.R_b, 1)}")

        # log info on daily-mean sea surface temp and daily-mean mixed layer depth
        day_log.log_sst_mld(col.tprofile[0], col.ml_thickness, n*col.dt)
        # log the processed tprofile
        col.log_tprofile()

        if (n+1) % plotstep == 0: # if timestep is multiple of plotstep, call plotter function
            plotcolumn(col, n, forcings, interval, depthlimit, ts_source="csv", obs_profile_source=source_file_dict,
                    show_initial=show_initial, show_obs=show_obs)

    # write to netcdf
    host_dir = r"D:\UKuni\3rdYr\MEP\cw\nc\noFW"
    nc_path = nc_TG.make_ncpath(host_dir, "COARE", dates, nctag)
    notes = "no freshwater, ml_method = dd"
    ts_series = forcings.getseries("tsvals")
    nc_TG.write_nc(nc_path, ts_series, col.zs, col.trans_tprofiles[:,1:], col.dt, day_log, notes)

# set wind_as_vel=True if wind forcing data is stored as velocities; False if it's stored as momentum fluxes
# ml_method determines the method used to find the mixed layer base index (and thus thickness, bulk Richardson
# number etc.). If ml_method == "dd_by_dz", these are calculated in terms of density gradient (ddensity/dz).
# If ml_method == "dd_from_surface", they are calculated by density-at-level minus density-at-surface.
# ml_method == "dt_from_surface" -> temperature criterion (0.2 or 0.5 deg C, set in parameters)
# ml_method == "dsigma_from_surface" -> potential density criterion (0.125 kg m^-3)
# m_show (mixings to show) is a list of some combination of "bulk", "convective" and "gradient", or "all", to determine
# what info is read out at each iteration while the model runs.
# Set mode = "test" to see a more detailed readout in the console of what step is being carried out at each moment
# dates can be None (default value) or a list of min,max dates formatted ["dd/mm/yyyy","dd/mm/yyyy"]. These will then
# be the min and max date values (inclusive) taken from the forcings csv.
# plotstep determines how frequently plots are presented. If plotstep == 1, profile plots will be shown at every loop.
# If plotstep == 10, plots will be shown every ten loops, etc. Default is None, which -> plots only shown on conclusion of
# model run
dates = ["23/10/1992", "3/3/1993"]
m_show = ["all"]

# run model
run_pwp(obs_profile_source=source_file_dict, obs_format="netcdf", forcings_csv=coare_forcings_csv, depthlimit=100,
        interval=0.01, plotstep=None, dates=dates, wind_as_vel=False, ml_method="dt_from_surface",
        m_show=m_show, mode="run", recordingstyle="rate", show_initial=True, show_obs=True, nctag="DT02noFW")
```

7.8.2 Column

This is the column class that stores the properties and functions of the water column

```
import numpy as np
import csv
from math import exp, cos, sin
import gsw  # https://github.com/TEOS-10/GSW-Python
from jdcal import gcal2jd
# custom profile classes, not included here
from tprofile import TProfile
from sprofile import SProfile
from dprofile import DProfile
from currentprofile import CurrentProfile
# various custom functions, not included here
import mixers
from parameters import param_dict
import absorption
from fetcher import getProfile, find_matching_jday


class Column:
    def __init__(self, obs_profile_source, dates, obs_format):
        self.mode="run"
        if obs_format == "csv":
            print("running csv section of if statement")
            with open(obs_profile_source) as f:  # read in data
                data = list(csv.reader(f))
            headers = data[0]          # first row of csv is headers Z, S, T, U, V for depth, sal, temp, currents
            data = data[1:]             # remaining rows are data
            zidx = headers.index("Z")
            sidx = headers.index("S")
            tidx = headers.index("T")
            uidx = headers.index("U")
            vidx = headers.index("V")
            self.nlevels = len(data)
            self.make_zero_profiles()         # make profiles with zeros as placeholders
            for level in range(self.nlevels):  # populate profiles with values from source file
                row = data[level]
                self.sprofile[level] = float(row[sidx])  # all wrapped in floats in case data stored as str
                self.tprofile[level] = float(row[tidx])
                self.uprofile[level] = float(row[uidx])
                self.vprofile[level] = float(row[vidx])
                self.zs[level] = float(row[zidx])

        if obs_format == "netcdf":
            source_file_dict = obs_profile_source
            profiledate_greg = dates[0]                                  # gregorian date as string dd/mm/yyyy
            day_greg, month_greg, year_greg = profiledate_greg.split("/")       # get day, month, year values
            profiledate_jd = sum(gcal2jd(year_greg, month_greg, day_greg))+0.5  # convert to Julian Day
            profile_idx = find_matching_jday(source_file_dict, profiledate_jd)  # find first match in jdays of profiles
            self.initial_profile_idx = profile_idx                       # log for later use
            zs, SVals = getProfile("sal", source_file_dict, record_number=profile_idx)[:2]  # get s profile, depth points
            TVals = getProfile("temp", source_file_dict, record_number=profile_idx)[1]      # get temp profile
            UVals = getProfile("east", source_file_dict, record_number=profile_idx)[1]      # get current profiles
            VVals = getProfile("north", source_file_dict, record_number=profile_idx)[1]
            self.nlevels = min(len(SVals), len(TVals), len(UVals))             # find the shortest profile
            for p in (zs, SVals, TVals, UVals, VVals):                       # truncate all profiles to this depth
                del p[self.nlevels:]
            self.zs = np.array([z - 0.5 for z in zs])  # profile fetcher was originally configured to return midpoints
            self.sprofile = SProfile(SVals)          # populate s and t profiles
            self.tprofile = TProfile(TVals)
            n_zeros = np.zeros(self.nlevels)
            self.uprofile = CurrentProfile(n_zeros) # populate current and density profiles with 0s
            self.vprofile = CurrentProfile(n_zeros)
            self.dprofile = DProfile(n_zeros)

        self.tprofiles=[self.tprofile]                # make a list where we store all the tprofiles for graphing
                                              # another tprofile will be produced at each model iteration
        self.trans_tprofiles = np.transpose(self.tprofiles)

        self.lat = param_dict["lat"]                  # lat attribute is set here bc needed for density calculations
        self.calculate_dprofile_gsw()

        self.initial_tprofile = self.tprofile.data.copy()       # log initial profiles for graphing
        self.initial_sprofile = self.sprofile.data.copy()
        self.initial_dprofile = self.dprofile.data.copy()

        self.dz = self.zs[1] - self.zs[0]             # extract dz from zs
        for profile in self:                         # set dz for all profiles (needed for e.g. fw flux on sprofile)
            profile.dz = self.dz
        self.dprofile.dz = self.dz                    # density done separately (see comment on __getitem__, below
        self.dt = param_dict["dt"]

    def make_zero_profiles(self):
        n_zeros = np.zeros(self.nlevels)               # construct empty profiles using relevant class constructors
        self.sprofile = SProfile(n_zeros)
        self.tprofile = TProfile(n_zeros)
        self.uprofile = CurrentProfile(n_zeros)
        self.vprofile = CurrentProfile(n_zeros)
        self.dprofile = DProfile(n_zeros)
        self.sigmaprofile = DProfile(n_zeros)
        self.zs = n_zeros

    # getter method to allow iteration over profiles in column. Doesn't include density because on most occasions when
    # we want to iterate over the profiles we don't want density- for mixing, e.g., we mix u, v, s and t and then
    # calculate density *from* s and t
    def __getitem__(self, position):
        profiles = [self.sprofile, self.tprofile, self.vprofile, self.uprofile]
        return profiles[position]

    # method to calculate densities using the gsw python package (Gibbs Seawater; TEOS 10)
    def calculate_dprofile_gsw(self, loop=0):
        if self.mode == "test":
            print("calculating density profile")
        ps = gsw.p_from_z(-1* self.zs, self.lat)
        sprofile = self.sprofile
        tprofile = self.tprofile
        dprofile = gsw.density.rho_t_exact(sprofile, tprofile, ps)
        self.dprofile[:] = dprofile

    def calculate_sigmaprofile(self):
        ps = gsw.p_from_z(-1 * self.zs, self.lat)
        p_ref = [1000]*len(ps)
        self.sigmaprofile = gsw.pot_rho_t_exact(self.sprofile, self.tprofile, ps, p_ref)

    # apply sensible, latent and terrestrial heat fluxes to top level of column
    def apply_sltq(self, heat_in, cp_sw = param_dict["cp_sw"]):   # specific heat capacity of seawater J kg^-1 K^-1
        if self.mode == "test":
            print("applying sltq")
        self.tprofile[0] += heat_in /(self.dz * self.dprofile[0] * cp_sw)

    def apply_solq(self, incident_solq):
        # I_lw(/sw) is fraction of solar radiation presumed to be longwave(/shortwave)
        if self.mode == "test":
            print("applying solq")
        n_levels = len(self.tprofile)
        for level in range(n_levels):
            dz = self.dz
            z = level*dz
            fraction_absorbed = absorption.fraction_absorbed(z, dz)
            q_in_at_level = incident_solq * fraction_absorbed
            self.tprofile[level] += (q_in_at_level /(dz * self.dprofile[0] * param_dict["cp_sw"]))

    # apply freshwater flux
    # sign convention: *all* fluxes are positive in the down (towards earth) direction
    # evaporation is positive down, i.e. usually negative
    # precipitation is positive down, i.e. usually positive
    # fw flux is ... = evap + precip (calculated prior to running model and input as a single forcing)
    # fw > 0 -> *gain* of freshwater -> *decrease* in salinity
    # salinity[0] -> salinity[0] * (1-fw/dz) (no dt term bc fw is accumulated over 1 hr)
    def apply_freshwater(self, fw):
        if self.mode == "test":
            print("applying fw")
        self.sprofile[0] *= (1-fw/self.dz)

    def mix_convective(self, mixings_to_show, cc_depth):
        if self.mode == "test":
            print("checking for convective instability")
        dprofile = self.dprofile
        stuvprofiles = [self.sprofile, self.tprofile, self.uprofile, self.vprofile] # could update. Column getter now excludes
density
        density_deltasigns = mixers.get_deltasigns(dprofile)
        while any(density_deltasign < 0 for density_deltasign in density_deltasigns):
            if self.mode == "test":
                print("found instability")
            last_neg_ddelt = mixers.get_last_neg(density_deltasigns)
            if any(val in mixings_to_show for val in ["convective", "all"]):
                print(f"convective mixing to {last_neg_ddelt + 1}")
            mixers.mix_to_n(stuvprofiles, last_neg_ddelt+1)
            self.calculate_dprofile_gsw()
            density_deltasigns = mixers.get_deltasigns(dprofile)
    def set_ml_base_idx(self, ml_method):
        if self.mode == "test":
            print("setting ml base index")
        dprofile = self.dprofile
        dds = np.diff(dprofile)  # make array of first differences of density
        ddbydzs = dds / self.dz  # divide by depth step to get change of density with depth between level
                                 # and next level
        if ml_method == "dd_by_dz":
            try:
                ml_base_idx = mixers.first_to_exceed(ddbydzs, param_dict["ml_ddbydz_threshold"])
                # ML base index = level at which density gradient first exceeds threshold...
            except:
                ml_base_idx = np.argmax(ddbydzs)
                # ...unless the gradient never exceeds that threshold. Then idx = level of maximum density gradient.
        elif ml_method == "dd_from_surface":
            dd_from_surface = abs(dprofile - dprofile[0])
            try:
                ml_base_idx = mixers.first_to_exceed(dd_from_surface, param_dict["ml_dd_from_surface_threshold"])
            except ValueError:
                ml_base_idx = np.argmax(dd_from_surface)
        elif ml_method == "dt_from_surface":
            tprofile = self.tprofile
            dt_from_surface = abs(tprofile - tprofile[0])
            ml_base_idx = mixers.first_to_exceed(dt_from_surface, param_dict["ml_dt_from_surface_threshold"])
        elif ml_method == "dsigma_from_surface":
            sigmaprofile = self.sigmaprofile
            dsigma_from_surface = abs(sigmaprofile - sigmaprofile[0])
            ml_base_idx = mixers.first_to_exceed(dsigma_from_surface,
param_dict["ml_dsigma_from_surface_threshold"])
        if ml_base_idx < 1:
            ml_base_idx = 1
        self.ml_base_idx = ml_base_idx
        self.ml_thickness = ml_base_idx * self.dz

    def rotate_by_angle(self, angle):
        if self.mode == "test":
            print("rotating")
        u = self.uprofile[:]
        v = self.vprofile[:]
        u2 = u * cos(angle) - v * sin(angle)
        v2 = u * sin(angle) + v * cos(angle)
        self.uprofile[:] = u2
        self.vprofile[:] = v2

    # pseudo code:
    # acceleration = windstress/(ml_thickness * average density)
    # can think of this as (force per unit area)/(mass per unit area); F=ma -> F/m = a
    # change in velocity dvel = acceleration*dt (dt in seconds)
    # vel += dvel for array down to ml_idx
    def apply_wind_stresses(self, wind_stresses_u_v):
        if self.mode == "test":
            print("applying winds")
        ml_base_idx = self.ml_base_idx
        ml_thickness = self.ml_thickness
        d_average_in_ml = np.mean(self.dprofile[:ml_base_idx])
        accelerations_u_v = [wind_stress/(ml_thickness * d_average_in_ml) for wind_stress in wind_stresses_u_v]
        [dvel_u, dvel_v] = [acceleration * param_dict["dt"] for acceleration in accelerations_u_v]
        self.uprofile[:ml_base_idx] += dvel_u
        self.vprofile[:ml_base_idx] += dvel_v

    def get_ml_delta(self, profile):
        ml_base_idx = self.ml_base_idx
        value_in_ml = profile[0]
        value_in_next_level = profile[ml_base_idx]
        ml_delta = value_in_next_level - value_in_ml
```

```python
        return ml_delta

    def set_bulk_richardson(self):
        # pseudo code:
        # R_b = (g * delta(density) * mixed_layer_thickness)/(reference density * delta(velocity)^2)
        # delta(var) = var(level below mixed layer) - var(mixed layer)
        if self.mode == "test":
            print("setting bulk Richardson number")
        ml_base_idx = self.ml_base_idx
        ml_thickness = self.ml_thickness
        ddensity = self.get_ml_delta(self.dprofile)
        du = self.get_ml_delta(self.uprofile)
        dv = self.get_ml_delta(self.vprofile)

        reference_density = param_dict["reference_density"]
        # reference_density = self.dprofile[0] # changed this as an experiment... deep bulk mixing is causing probs
        g = param_dict["g"]
        dvel_sq = du**2 + dv**2
        if ml_base_idx > self.nlevels - 2:
            R_b = 0.65
            # when spinning up, high currents can lead to bulk mixing down to base of column. In that case we set
            # R_b to 0.65 and continue. Otherwise it tries to mix off the bottom of the column.
        elif dvel_sq > 0:
            R_b = (g * ddensity * ml_thickness) / (reference_density * dvel_sq)
        else:
            R_b = np.inf
        self.R_b = R_b

    def bulk_r_mix(self, mixings_to_show, day_log, nudgepoint=None, nudgeval=0):
        if self.mode == "test":
            print("bulk richardson mixing")
        R_b_critical = 0.64 # 0.65 canonically; changed to avoid a specific over-mixing event
        i=0
        while self.R_b < R_b_critical:                              # while(/if) it's less than the critical value...
            if any(val in mixings_to_show for val in ["bulk", "all"]):
                print(f"R_b = {round(self.R_b,3)}; bulk mixing to {int(self.ml_base_idx * self.dz)}m")
            for profile in self:
                mixers.mix_to_n(profile, self.ml_base_idx)   # ... mix stuv profiles down to the base level...
            self.calculate_dprofile_gsw()                   # ...recalculate density...
            self.ml_base_idx += 1                           # ...add another level to the mixed layer to be mixed in next time..
            self.set_bulk_richardson()                      # and recalculate bulk richardson number
            if i == nudgepoint:
                day_log.nudges += 1                         # this logs how many times R_b has been nudged
                R_b_critical = nudgeval
                print("nudging R_b")
                # Bulk mixing occasionally entrains >80 layers in one loop
                # and cools the SST hugely in a way that doesn't square at all with observations
                # In my runs of ~one year of hourly data, it does this 0-1 times per run
                # This fudge relaxes stringency of the bulk stability criterion once we've mixed {nudgepoint} layers.
                # R_b_critical is reset when bulk_r_mix is next called
                # On my data, with nudgepoint = 10, this function was called on about 1 loop in 20, and raised MLD
                # by average 5m. This was a problem, so the nudgepoint was raised to 28. Can be set in main.
            i += 1

    # function to get gradient richardson number at a given level
    def get_gradient_richardson(self, j):
        # pseudo code:
        # R_g = (g * dd * dz) / (d_ref * (du^2 + dv^2)) where d{var} = var_(j+1) - var_j
        g = param_dict["g"]
        d_ref = param_dict["reference_density"]
        dd = self.dprofile[j+1] - self.dprofile[j]
        dz = self.dz
        du = self.uprofile[j+1] - self.uprofile[j]
        dv = self.vprofile[j+1] - self.vprofile[j]
        if du*dv == 0:
            R_g = np.inf
    else:
        R_g = (g * abs(dd) * dz) / (d_ref * (du**2 + dv**2))
        # abs val of dd has been taken here as a patch. When processing 13/06/95, AS data, some dd came back
        # negative, -> R_g < 0, and it couldn't be resolved
    return R_g

# set the gradient richardson numbers for each level in the column (except the bottom one)
def set_grprofile(self):
    if self.mode == "test":
        print("calculating gradient richardson profile")
    nlevels = self.nlevels
    self.grprofile = np.empty(nlevels-1)
    for j in range(nlevels - 1):
        self.grprofile[j] = self.get_gradient_richardson(j)

# scan gr profile
# if min value is below critical level, apply gradient richardson mixing
def gr_mix_col(self, mixings_to_show, day_log):
    if self.mode == "test":
        print("gradient richardson mixing")
    min_gr = min(self.grprofile)
    i = 0
    while min_gr < 0.25 and i < 1000:
        min_gr_index = np.argmin(self.grprofile)          # this level and the one below are mixed
        if any(val in mixings_to_show for val in ["gradient", "all"]):
            print(f"min. R_g = {round(min_gr,3)}; gradient mixing at {int(min_gr_index*self.dz)}m")
        for profile in self:
            profile.gr_mix_profile(min_gr_index, min_gr)
        self.calculate_dprofile_gsw()
        gr_mix_min_idx = max(min_gr_index-1,0)            # these four layers are recalculated
        gr_mix_max_idx = min(min_gr_index+2, self.nlevels-1)
        for j in range(gr_mix_min_idx, gr_mix_max_idx):
            self.grprofile[j] = self.get_gradient_richardson(j)
        min_gr = min(self.grprofile)
        i += 1          # to prevent endless loops.
        if i == 1000:
            day_log.gradmaxes += 1                        # add 1 to record of times gr mixing has maxed out
            print("gradient Richardson mixing maxed out")

# we log both tprofiles and transposed tprofiles. A tprofile has temps for each depth at a given time:
# [T_(n=n, z=0), T_(n=n, z=1)... T_(n=n, z=max.depth)]
# a transposed t profile has all temps generated so far for a particular depth:
# [T_(n=0,z=z), T_(n=1, z=z)... T_(n=latest timestep, z=z)]
# transposed tprofiles are used for heatmapping and recording to csv.
def log_tprofile(self):
    if self.mode == "test":
        print("logging t profiles")
    tprofile_to_append = (self.tprofile.data).copy()                  # make a copy of the profile
    self.tprofiles.append(tprofile_to_append)                         # append to profiles
    trans_tprofile_to_append = np.transpose([tprofile_to_append])      # transpose
    self.trans_tprofiles = np.append(self.trans_tprofiles, trans_tprofile_to_append, axis=1)    # append to array
```