## UNIFIED MODEL DOCUMENTATION PAPER NO S1

# INTERPOLATION TECHNIQUES AND GRID TRANSFORMATION USED IN THE UNIFIED MODEL

by

D.M Goddard

Version 13

23rd October 1998

Model Version 4.5

Numerical Weather Prediction
Meteorological Office
London Road
BRACKNELL
Berkshire
RG12 2SZ
United Kingdom

| Modification Record | | |
|---|---|---|
| **Model version** | **Author** | **Description...................** |
| 1.11 | | Appendix 1: H_INT_CO argument list changed to accommodate non-uniform grids. V_INT_Z changed to ignore top model level.<br>Appendix 2:TRACER_LEVELS_ADV_OUT added to namelist SIZE.<br>Use of unit 19 changed. Unit 37 no longer used. |
| 1.12 | | Appendix 2:Sample code to access ECMWF analyses changed to UNICOS |
| 1.15 | | Appendix 2:Logical RESET added to Namelist SIZE |
| 1.17 | | Method of specifying topography subarea revised STRAT_Q introduced to Namelist SIZE |
| 2.2 | | Appendix 2:Option to use MSLP in ECMWF grib code extraction introduced.<br>Order of MARS upper air extraction changed. |
| 2.5 | | Appendix 2:Ocean dumps with no bottom topography can be reconfigured.<br>MARS extraction example changed because of RACF changes |
| 2.7 | | Appendix 2:Option to initialise tracer variables introduced |
| 3.1 | | Appendix 2:<br>a ECMWF grib code interface extended to include access to date stored on ECMWF model levels. Grib interface made portable.<br>b.Transplant of data from one dump into another supported<br>c.Incorporation of ECMWF ensemble perturbations supported. |
| **Document version** | | |
| Version 8 | A Dickinson | Equation 3.6a new. Equation 3.16 modified.<br>Equation 3.18 corrected. |
| Version 9 | | Appendix 2 revised |
| Version 10 | D. Goddard | Appendix 2: new routines INTF_COAST_AJ and SPIRAL_S added. corrections to other argument lists. |
| Version 10.1 | | Equation 3.9 corrected, $T_S$ replaces $T_*$ |
| Version 11 | D.M Goddard | New option for vertical interpolation is described for output modelling. in section 3.3.1 and appendix 3.<br>Appendices 1 and 2 updated for UM version 4.3 and MPP |
| Version 12 | D.M. Goddard | Appendix 2 updated for UM version 4.4 |
| Version 13 | D.M. Goddard | Appendices 1 and 2 updated for UM version 4.5.<br>Equations corrected in section 5.1. |

## 1. INTRODUCTION

Interpolation is the mechanism by which data is transformed from one grid to another. As such it is present in all parts of the unified model suite. The analysis scheme, for example, is a specific application of interpolation. This note concentrates on the techniques used in two areas of the model in particular:

(i) the interpolation of initial analyses and lateral boundary data to new resolutions
(ii) the interpolation of primary model variables to pressure levels for output.

Horizontal interpolation is discussed in Chapter 2 and vertical interpolation in Chapter 3. Chapter 4 describes the techniques used for temporal interpolation of ancillary fields. The transformation between the latitude longitude grid and the equatorial latitude longitude grid used in regional versions of the model is described in Chapter 5. One of the main applications of interpolation is the provision of initial model fields at new resolutions through the reconfiguration step. All the interpolation components necessary to achieve this are described in the document. The reconfiguration step is described in Appendix 2.

## 2. HORIZONTAL INTERPOLATION

### 2.1 General formula

Horizontal interpolation is carried out using a standard bi-linear technique. Assuming a regular x-y grid of resolution $(\Delta x, \Delta y)$, an intermediate value of field F at coordinates $(x', y')$ is estimated from the formula

$$\begin{aligned} F(x',y') = {} & (1-a)(1-b)F(x,y) + a(1-b)F(x+\Delta x,y) + b(1-a)F(x,y+\Delta y) \\ & + ab\,F(x+\Delta x,y+\Delta y) \end{aligned} \tag{2.1}$$

where $x \geq x' \succ x + \Delta x$ and $y \geq y' \succ y + \Delta y$. The weights a and b are given by

$$a = \frac{x'-x}{\Delta x} \text{ and } b = \frac{y'-y}{\Delta y} \tag{2.2}$$

A special case of this formula, widely used in the discretization of the integration scheme, is the calculation of the value at the centre of a grid box. In this case a=b=0.5.

### 2.2 Coastal adjustment

Coastal values of surface temperature, soil moisture content and other surface fields require further adjustment after the application of equation (2.1). This is because horizontal interpolation in these regions uses both land and sea values which can generate inappropriate results. The adjustment is controlled by a land-sea mask at the new resolution, coastal values on the new grid being reset to the value of the nearest point of the same type on the old grid.

Interpolation to higher resolutions may resolve islands or lakes which are several grid lengths away from other grid points of the same type. The search radius is limited to one gridlength to ensure that representative values are used. If no point of the same type exists within this radius, then the new point is assigned the value (but not the type) of the nearest point of the opposite type on the old grid. Separate pointers to unresolved sea and land points are provided so that corrections (say using climatology) may be made by the user.

## 3. VERTICAL INTERPOLATION

The requirements for vertical interpolation are threefold; from hybrid to pressure coordinates for output purposes; from hybrid to hybrid coordinates to allow a redistribution or change in the number of model levels; and finally, from pressure to hybrid coordinates to support the importing of analyses on pressure levels from other centres or from the WGDOS archive.

The hybrid coordinate, $\eta$, is defined by

$$\eta = A + B \tag{3.1}$$

where

$$p = Ap_0 + Bp_* \tag{3.2}$$

Vertical interpolation therefore reduces to an interpolation between two arbitrary sets of pressure levels.

### 3.1 General formula

Vertical interpolation of u and v, relative (rather than specific) humidity and cloud liquid water is carried out assuming that these quantities vary linearly with log pressure. The value of field F at pressure $p_i$ is given by the formula:

$$F_i = \begin{cases} F_{top} & p_i < p_{top} \\ \alpha F_j + (1-\alpha)F_{j-1} & p_j < p_i < p_{j-1} \\ F_1 & p_i > p_1 \end{cases} \tag{3.3}$$

where j and j-1 are levels at which the field F is known immediately above and below level i, and

$$\alpha = \frac{\ln\left[\dfrac{p_i}{p_{j-1}}\right]}{\ln\left[\dfrac{p_j}{p_{j-1}}\right]} \tag{3.4}$$

When outputting wind fields, an extra level of winds derived through maximum wind modelling is used.

3.2 Height and mean sea level pressure

The height field is required for output purposes. The method used is that proposed by Swinbank and Wilson (1990) and is based on the hydrostatic relationship. It differs from the calculation of height in the dynamics in several ways with both the correction term which ensures angular momentum conservation and the additional Coriolis terms being ignored.

From the hydrostatic equation the height of each $\eta$ layer boundary (Figure 1) is given by

$$z_{k-\frac{1}{2}} = z_* + \frac{c_p}{g} \sum_{m=1}^{m=k-1} \left(\theta_v\right)_m \Delta\pi_m \tag{3.5}$$

where k = 2,....,top. The height at pressure $p_i$ within layer k is then calculated using a second order approximation to the thickness of the layer between $p_i$ and $p_{k-\frac{1}{2}}$

$$z_i = z_{k-\frac{1}{2}} + \frac{c_p}{g}\left\{(\theta_v)_k\left[\pi_{k-\frac{1}{2}} - \pi_i\right] - \frac{1}{2}\left(\frac{\partial\theta}{\partial\pi}\right)_k\left[\pi_i(\pi_i - 2\pi_k) - \pi_{k-\frac{1}{2}}(\pi_{k-\frac{1}{2}} - 2\pi_k)\right]\right\} \tag{3.6}$$

$$\pi_k = \frac{\left[\pi_{k+\frac{1}{2}}\, p_{k+\frac{1}{2}} - \pi_{k-\frac{1}{2}}\, p_{k-\frac{1}{2}}\right]}{(\kappa+1)\left[p_{k+\frac{1}{2}} - p_{k-\frac{1}{2}}\right]} \tag{3.6a}$$

In the above $\pi=(p/100000)^{\kappa}$ for $\pi_i, \pi_{k+\frac{1}{2}}, \pi_{k-\frac{1}{2}}$ and $\theta_v$ is the virtual potential temperature defined by $\theta_v = T(1+0.61q)\pi^{-1}$ . $\kappa = R/c_p$ , where R is the gas constant and $c_p$ is the specific heat capacity of dry air. The full model level value of $\pi_k$ is consistent with the geopotential equation (UM Documentation paper 10, equation 26) and is given by

The local gradient $\left(\frac{\partial\theta}{\partial\pi}\right)_k$ is defined as follows

$$\left(\frac{\partial\theta}{\partial\pi}\right)_k = \left\{\frac{T_{kp} - T_{km}}{\pi_{kp} - \pi_{km}} - \theta_k\right\}\pi_k^{-1} \qquad (3.7)$$

where km = max(1,k-1) and kp = min(top,k+1). This formulation is designed to give an accurate representation of $\left(\frac{\partial\theta}{\partial\pi}\right)_k$ for an isothermal atmosphere; this is particularly important in the stratosphere where the model layers are very deep.

The height of a pressure level above the top model layer is calculated by using equation (3.6) with k=top.

When calculating the height of a pressure level below the surface of the earth, that is when $p_i > p_*$ , a constant lapse rate $\gamma = 0.0065^{\circ}\mathrm{Km}^{-1}$ is assumed from a reference temperature outside the model's boundary layer. Using the standard altimeter equation, the temperature at the surface is given by

$$T_s = T_r \left( \frac{p_*}{p_r} \right)^{\frac{\gamma R}{g}} \tag{3.8}$$

where r=5 (when there are 4 levels in the boundary layer). The extrapolated temperature at level $p_i$ is given by a similar expression and can also be calculated from

$$T_i = T_s + \gamma(z_* - z_i) \tag{3.9}$$

It then follows that the height of level $p_i$ is given by

$$z_i = z_* + \left( \frac{T_s}{\gamma} \right) \left[ 1 - \left( \frac{p_i}{p_*} \right)^{\frac{\gamma R}{g}} \right] \tag{3.10}$$

Mean sea level pressure may be calculated by substituting equation (3.8) into equation (3.9) and setting $z_i = 0$ :

$$p_{msl} = p_* \left( \frac{T_s + \gamma z_*}{T_s} \right)^{\frac{g}{\gamma R}} \tag{3.11}$$

## 3.3 Temperature

### 3.3.1 Output modelling

The vertical interpolation of temperature for output purposes follows Swinbank and Wilson (1990) and is based on the assumption that T varies linearly with geopotential height. The temperature at pressure $p_i$ is calculated in one of two ways depending on whether $p_i$ falls in the top or bottom half of layer k:

$$T_i = \begin{cases} T_k + \dfrac{\left(T_{k-1} - T_k\right)\theta_k\left(\pi_i - \pi_k\right)}{\theta_k\left(\pi_{k-\frac{1}{2}} - \pi_k\right) + \theta_{k-1}\left(\pi_{k-1} - \pi_{k-\frac{1}{2}}\right)} & \pi_{k-\frac{1}{2}} \geq \pi_i \succ \pi_k \\ \\ T_k + \dfrac{\left(T_{k+1} - T_k\right)\theta_k\left(\pi_k - \pi_i\right)}{\theta_k\left(\pi_k - \pi_{k+\frac{1}{2}}\right) + \theta_{k+1}\left(\pi_{k+\frac{1}{2}} - \pi_{k+1}\right)} & \pi_k \geq \pi_i \succ \pi_{k+\frac{1}{2}} \end{cases} \tag{3.12}$$

Extrapolation is performed above the top level as follows:

$$T_i = T_{top} + \frac{\left(T_{top} - T_{top-1}\right)\theta_{top}\left(\pi_{top} - \pi_i\right)}{\theta_{top}\left(\pi_{top-\frac{1}{2}} - \pi_{top}\right) + \theta_{top-1}\left(\pi_{top-1} - \pi_{top-\frac{1}{2}}\right)} \qquad \pi_i \preceq \pi_{top} \tag{3.13}$$

and between the bottom level and the surface as follows:

$$T_i = T_1 + \frac{\left(T_1 - T_2\right)\theta_1\left(\pi_i - \pi_1\right)}{\theta_2\left(\pi_{1\frac{1}{2}} - \pi_2\right) + \theta_1\left(\pi_1 - \pi_{1\frac{1}{2}}\right)} \qquad \pi_* \preceq \pi_i \succ \pi_1 \tag{3.14}$$

$$T_i = T_r\left[\frac{p_i}{p_r}\right]^{\frac{\gamma R}{g}} \qquad p_i \succ p_* \tag{3.15}$$

Below the surface the standard altimeter equation is used as in the calculation of height (see equation (3.8)) with r=2.

To use equations 3.12 to 3.15 the model variable, potential temperature, $\theta_k$ has to be converted to temperature $T_k$ (= $\Pi_k\ \theta_k$). All conversions internal to the model are done using the specification of $\Pi_k$ given by equation (3.6a).

At stratospheric levels with thick model layers this can lead to a warm bias in derived temperatures (see Appendix 3). An ***optional*** vertical interpolation which uses an alternative specification of $\Pi_k$ is available for ***output modelling*** and processing of satellite soundings (GLOSS/LASS). The specification for $\Pi_k$ is designed to give zero error for an isothermal layer, $T_k$ , whose geopotential thickness is

$$\Delta\Phi = R\ T_k\ \ln(p_{k-1/2}/p_{k+1/2}) = c_p\ \theta_k\ (\Pi_{k-1/2} - \Pi_{k+1/2})$$

so that, $T = \theta_k\ \Pi_k^{\#}$

which leads to the nominal model level value for $\Pi_k^{\#}$ given by

$$\Pi_k^{\#} = \kappa \frac{\Pi_{k-1/2} - \Pi_{k+1/2}}{\ln\ (p_{k-1/2}\ /\ p_{k+1/2})} \qquad (3.6b)$$

This optional conversion is used for output *on pressure levels* if requested by the logical switch QV_INT_TP =.TRUE., which is included in the operational modification sets. The default is .FALSE., so that climate and other users continue to use the expression given by (3.6a).

### 3.3.2 Tropopause modelling

The calculation of the temperature, pressure and height of the tropopause follows Swinbank and Wilson (1990). First the lapse rates between model levels are calculated from

$$\gamma_{k+\frac{1}{2}} = \frac{T_k - T_{k+1}}{z_{k+1} - z_k} = \frac{g\left(T_k - T_{k+1}\right)}{c_p\left(\theta_{k+1}(\pi_{k+\frac{1}{2}} - \pi_{k+1}) + \theta_k(\pi_k - \pi_{k+\frac{1}{2}})\right)} \qquad (3.16)$$

A search is made for the first lapse rate less than $0.002^{\circ}\mathrm{Km}^{-1}$ which is above 500mb and with the next lapse rate above also less than $0.002^{\circ}\mathrm{Km}^{-1}$. If this is found at level $j-\frac{1}{2}$, then the tropopause is between level j-1 and level j. The tropopause temperature $T_T$ is then given by simultaneously extrapolating from $T_j$ and $T_{j-1}$ using the lapse rates $\gamma_{j+\frac{1}{2}}$ and $\gamma_{j-1\frac{1}{2}}$ respectively for the layers above and below the tropopause layer:

$$T_T = T_{j-1} - \gamma_{j-1\frac{1}{2}}\left(z_T - z_{j-1}\right) = T_{j-1} - \gamma_{j-1\frac{1}{2}}\left( z_T - z_{j-\frac{1}{2}} - \left( z_{j-1} - z_{j-\frac{1}{2}} \right) \right) \qquad (3.17)$$

$$T_T = T_j - \gamma_{j+\frac{1}{2}}\left(z_T - z_j\right) = T_j - \gamma_{j+\frac{1}{2}}\left( z_T - z_{j-\frac{1}{2}} - \left( z_j - z_{j-\frac{1}{2}} \right) \right) \qquad (3.18)$$

For simplicity and consistency with equation (3.16), the tropopause calculation neglects the second order term used in equation (3.6) so that

$$z_{j-i} - z_{j+\frac{1}{2}} = c_p/g\ \theta_{j-1}(\pi_{j-\frac{1}{2}} - \pi_{j-1}) \qquad (3.18a)$$

$$z_j - z_{j-\frac{1}{2}} = c_p/g\ \theta_j\ (\pi_{j-\frac{1}{2}} - \pi_j) \qquad (3.18b)$$

with $\pi_j$ , $\pi_{j-1}$ calculated using equation (3.6a). Equations (3.17) and (3.18) may be solved for $z_T - z_{j-\frac{1}{2}}$ to give

$$z_T - z_{j-\frac{1}{2}} = \frac{T_{j-1} - T_j + \gamma_{j-1\frac{1}{2}}\left(z_{j-1} - z_{j-\frac{1}{2}}\right) - \gamma_{j+\frac{1}{2}}\left(z_j - z_{j-\frac{1}{2}}\right)}{\gamma_{j-1\frac{1}{2}} - \gamma_{j+\frac{1}{2}}} \tag{3.19}$$

with the proviso that $-\frac{1}{2}\Delta z_{j-1} \preceq z_T - z_{j-\frac{1}{2}} \preceq \frac{1}{2}\Delta z_j$. The temperature, $T_T$, and pressure, $p_T$,

of the tropopause may then be found from:

$$T_T = T_j - \gamma_{j+\frac{1}{2}}\left(z_T - z_j\right) \tag{3.20}$$

$$p_T = p_0 \pi_T^{1/\kappa} \tag{3.21}$$

where

$$\pi_T = \begin{cases} \pi_{j-\frac{1}{2}} - \dfrac{g}{c_p \Theta_{j-1}} \left\{z_T - z_{j-\frac{1}{2}}\right\} & z_T - z_{j-\frac{1}{2}} \preceq 0 \\ \pi_{j-\frac{1}{2}} - \dfrac{g}{c_p \Theta_j} \left\{z_T - z_{j-\frac{1}{2}}\right\} & z_T - z_{j-\frac{1}{2}} \succ 0 \end{cases} \tag{3.22}$$

### 3.3.3 Importing temperature data

At version 2.7, the technique used to vertically interpolate temperature data imported on pressure surfaces or non standard *n*-surfaces into the model was changed to use the general formulae given in section 3.1, i.e. it assumes temperature varies linearly with log p. Conversion to potential temperature, $\Theta_k = T_k/\pi_k$, is done with $\pi_k$ calculated as in equation (3.6a). The change from the previous method described below was made because that method was found to give large errors when interpolating between two sets of levels of equal number with close location of corresponding levels.

Previous to version 2.7, the technique used to vertically interpolate temperature data imported on pressure surfaces or non standard *n*-surfaces into the model was based on the ideas of Swinbank and

Wilson (1990). The thickness of a model layer is first calculated by summing the partial thicknesses between adjacent model half levels and the intervening input levels. The layer thickness is then converted directly to potential temperature.

Given an input temperature profile $T_i$ varying with pressure $p_i$, a function $T_p$ may be constructed such that $T$ varies linearly with log $p$ between input levels. The thickness $\Delta\phi$ between any two pressure levels $p_a$ and $p_b$ is given by

$$\Delta\phi = \int_{p_a}^{p_b} R\ T\ d(\log p) \tag{3.23}$$

If $p_a$ and $p_b$ are chosen to be model layer boundaries (say $p_{k+\frac{1}{2}}$ and $p_{k-\frac{1}{2}}$) the layer potential temperature $\Theta_k$ can be calculated in a manner consistent with the model hydrostatic equation:

$$\Delta\phi_k = c_p \Theta_k \left( \pi_{k-\frac{1}{2}} - \pi_{k+\frac{1}{2}} \right) \tag{3.24}$$

Thus

$$\Theta_k = \int_{p_{k-\frac{1}{2}}}^{p_{k+\frac{1}{2}}} \frac{\kappa\ T}{\left( \pi_{k-\frac{1}{2}} - \pi_{k+\frac{1}{2}} \right)}\ d(\log p) \tag{3.25}$$

In order to use equation (3.25) it is necessary to calculate $T_{k+\frac{1}{2}}$ and $T_{k-\frac{1}{2}}$. Different calculations are required depending on the position of a layer relative to the distribution of input levels. When the input levels cover a model layer, that is when

$$p_{k-\frac{1}{2}} \preceq p_1 \text{ (input) and } p_{k+\frac{1}{2}} \succeq p_{top} \text{ (input),}$$

the temperature at a layer boundary $T_b \; ( = T_{k+\frac{1}{2}} \text{ or } T_{k-\frac{1}{2}})$ is calculated from

$$T_b = T_i + \left(T_{i+1} - T_i\right) \frac{\ln\left[p_b/p_i\right]}{\ln\left[p_{i+1}/p_i\right]} \tag{3.26}$$

where i is the index of the input level immediately below and i+1 the index of the input level immediately above level b.

If the input data only gives partial coverage of a layer, that is at the bottom of the atmosphere when

$$p_{k-\frac{1}{2}} \succ p_1 \text{ (input) and } p_{k+\frac{1}{2}} \prec p_1 \; ,$$

or at the top of the atmosphere when

$$p_{k+\frac{1}{2}} \prec p_{top} \text{ (input) and } p_{k-\frac{1}{2}} \succ p_{top} \text{ (input),}$$

the temperature at the layer boundary outside the area of data coverage is calculated by extrapolation from the two adjacent input levels using the assumption that T varies linearly with log p.

The layer mean value of $\Theta$ is then calculated as follows

$$\Theta_k = \kappa \Bigg\{ \left( T_{k+\frac{1}{2}} + T_{i+j} \right) \ln\left[ p_{i+j}/p_{k+\frac{1}{2}} \right] + \sum_{m=0}^{m=j-1} \left( T_{i+m} + T_{i+m+1} \right) \ln\left[ p_{i+1}/p_{i+I+1} \right]$$

$$+ \left( T_{k-\frac{1}{2}} + T_i \right) \ln\left[ p_{k-\frac{1}{2}}/p_i \right] \Bigg\} \Big/ 2 \left( \pi_{k-\frac{1}{2}} - \pi_{k+\frac{1}{2}} \right) \tag{3.27}$$

where i is the index of the first input level above level $k-\frac{1}{2}$ and i+j is the index of the input level

immediately below level $k+\frac{1}{2}$ .

If a model layer falls completely above the top level of the input data, then the temperature of the layer is assigned the value of the topmost input level. Thus

$$\Theta_k = T_{top}\pi_k^{-1} \quad \text{if} \quad p_{k-\frac{1}{2}} \prec p_{top}\,(\text{input}) \tag{3.28}$$

If a model layer is very thick (in terms of log p), equation 3.27 can generate an interpolated temperature value which is outside the range of temperature values on the input levels; see Swinbank and Wilson (1990) for a discussion of this type of interpolation error. If the top model layer straddles the top input level a check is made to ensure that the top output temperature is between the top two input temperatures; if this is not the case equation 3.28 is applied.

At the bottom of the atmosphere equation (3.15) is used with r=1:

$$\Theta_k = T_1\pi^{-1}\left[\frac{p_k}{p_1}\right]^{\frac{\gamma R}{g}} \quad \text{if} \quad p_{k+\frac{1}{2}} \succ p_1\,(\text{input}) \tag{3.29}$$

## 4. TEMPORAL INTERPOLATION

Time dependent ancillary fields are available at monthly intervals. Temporal interpolation is therefore required to provide values at intermediate times.

In the simplest case, linear interpolation is used as follows to estimate the value of a field F at time t:

$$F(t) = \frac{(t_2-t)}{(t_2-t_1)}F(t_1) + \frac{(t-t_1)}{(t_2-t_1)}F(t_2) \tag{4.1}$$

where $F(t_1)$ and $F(t_2)$ are known and $t_1 \le t \le t_2$ .

The temporal interpolation of snow depth, sea-ice surface temperature, sea-ice fraction, sea-ice thickness and SST are complicated by interdependences with other fields. For example, the melting of sea ice in the period $t_1 - t_2$ switches the definition of a grid point from sea-ice to sea. Some knowledge of the time at which the transition takes place must be provided in order that sensible decisions can be made about the time interpolation of both SST and sea-ice temperature.

For the above fields (denoted by F) linear interpolation in time is used, except where a controlling field, $\varsigma$ , changes from zero to non-zero. For sea fields $\varsigma$ is sea-ice fraction; for land fields $\varsigma$ is snow depth. The fractional time, $\tau$ , between $t_1$ and $t_2$ at which this change occurs for each point is supplied. A prescribed value, $F_p$ , is inserted where $\varsigma$ is zero. F is linearly interpolated between its value at the time when $\varsigma$ is non-zero and the prescribed value when $\varsigma$ becomes zero. If t is first converted into the fractional time between $t_1$ and $t_2$ ,

$$\alpha = \frac{t-t_1}{t_2-t_1} \qquad (4.2)$$

then the procedure adopted when $\varsigma$ changes from zero to non-zero may be summarised algebraically as follows:

$$F(t) = \begin{cases} F_p & \alpha \prec \tau \text{ and } \varsigma(t_1) = 0 \\ & \alpha \succ \tau \text{ and } \varsigma(t_2) = 0 \\ \dfrac{(1-\alpha)}{(1-\tau)} F_p + \dfrac{(\alpha-\tau)}{(1-\tau)} F(t_2) & \alpha \geq \tau \text{ and } \varsigma(t_1) = 0 \\ \dfrac{(\tau-\alpha)}{\tau} F(t_1) + \dfrac{\alpha}{\tau} F_p & \alpha \leq \tau \text{ and } \varsigma(t_2) = 0 \end{cases} \tag{4.3}$$

The fractional time at which the field $\varsigma$ changes from zero to non-zero or visa versa is calculated as follows:

(i) In order to simplify the procedure, each line of longitude is treated separately. A positive indicator is set at points which have a zero value at time $t_1$ and a non-zero value at time $t_2$, and a negative indicator at points which have a non-zero value at $t_1$ and a zero value at $t_2$. A zero indicator is set at all other points.

(ii) The fractional time at which each of these points changes from zero to non-zero is calculated by searching for consecutive groups of either positive or negative indicators enclosed by zero indicators. Where such a group has a common value to the south of $\varsigma_s$ and to the north of $\varsigma_n$, with one of $\varsigma_s$ or $\varsigma_n$ zero and the other non-zero, the time of becoming zero within the group is set to change in a linear way across the group. In all other cases a simple progression of the snow or ice edge cannot be safely assumed and the fractional time is set to 0.5.

## 5. EQUATORIAL LATITUDE-LONGITUDE GRID

Regional versions of the unified model use a latitude-longitude grid in which the polar singularities, usually situated at the celestial poles, are translated along a meridian so that an equatorial latitude-longitude grid covers the integration area. Such a grid has the advantage of being more or less uniform over the whole of the integration area and avoids the need for stability filtering in the forecast model at high latitudes. In the following $(\lambda,\phi)$ and $(\lambda',\phi')$ are latitude longitude coordinates on the standard and transformed grids respectively.

5.1 Transformation formulae

Relationships between the standard latitude-longitude grid and the equatorial grid may be derived through the use of vector algebra. Let $\underline{i}$ , $\underline{j}$ , $\underline{k}$ be unit vectors in Cartesian coordinates with $\underline{\hat{r}}$ , $\underline{\hat{\phi}}$ , $\underline{\hat{\lambda}}$ and $\underline{\hat{r}}$ , $\underline{\hat{\phi}}'$ , $\underline{\hat{\lambda}}'$ unit vectors in spherical polar coordinates on the latitude-longitude and equatorial grids respectively.

The unit radius vector, corresponding to a point $(\lambda, \phi)$ on the earth's surface, may be calculated from simple geometry (see Figure 2) and is given by

$$\underline{\hat{r}} = \cos\lambda\ \cos\phi\ \underline{i} + \sin\lambda\ \cos\phi\ \underline{j} + \sin\phi\ \underline{k} \tag{5.1}$$

It may be assumed that the latitude of the new pole, $\phi_p$ , lies somewhere on the 180˚E meridian.

This assumption is not restrictive and results may be easily generalised for translation of the pole along any meridian by first rotating the standard latitude-longitude grid through a longitudinal angle $\lambda_o$ , where $\lambda_o=\lambda_p+\pi$ . The radius vector of the new pole is then given by

$$\underline{\hat{r}_p} = -\cos\phi_p\ \underline{i} + \sin\phi_p\ \underline{k} \tag{5.2}$$

The transformation is equivalent to rotating the x-z plane about the y axis by an angle

$\pi/2-\phi_p$ . $\underline{\hat{r}}_{\underline{p}}$ is therefore normal to the equatorial plane of the new grid which contains the y axis (see Figure 3). The latitude of $\underline{\hat{r}}$ in the new system, $\phi'$ , can be found from the scalar product of vectors $\underline{\hat{r}}$ and $\underline{\hat{r}}_{\underline{p}}$ . It is given by

$$\phi' = \sin^{-1}(\underline{\hat{r}}.\underline{\hat{r}}_{\underline{p}}) \tag{5.3}$$

On substituting for $\underline{\hat{r}}$ and $\underline{\hat{r}}_{\underline{p}}$ this becomes

$$\phi' = \sin^{-1}\left(-\cos\phi_p\cos(\lambda-\lambda_o)\cos\phi + \sin\phi_p\sin\phi\right) \tag{5.4}$$

The vector product $\underline{\hat{r}}$ x $\underline{\hat{r}}_{\underline{p}}$ can be used to obtain the new longitude $\lambda'$ . It also lies on the equatorial plane of the new grid and is perpendicular to the projection of $\underline{\hat{r}}$ onto this plane. The angle $\lambda'$ is then obtained through the scalar product of $\underline{\hat{r}}$ x $\underline{\hat{r}}_{\underline{p}}$ and $-\underline{j}'$. Thus

$$\lambda' = \cos^{-1}\left[\frac{-(\underline{\hat{r}}\times\underline{\hat{r}}_{\underline{p}})\cdot\underline{j'}}{|\,\underline{\hat{r}}\times\underline{\hat{r}}_{\underline{p}}\,|}\right] \tag{5.5}$$

On substituting for $\underline{\hat{r}}$ and $\underline{\hat{r}}_{\underline{p}}$ , we get

$$\lambda' = \gamma\cos^{-1}\left[\frac{\cos\phi_P\sin\phi + \sin\phi_P\cos(\lambda-\lambda_o)\cos\phi}{\cos\phi'}\right] \tag{5.6}$$

where from global symmetry $\gamma = 1$ if $0\le \lambda-\lambda_o\le\pi$ and $\gamma = -1$ if $-\pi\le \lambda-\lambda_o<0$ . Similar equations relating $\lambda$ and $\phi$ to $\lambda'$ and $\phi'$ may be obtained by reversing the above analysis. These are

$$\phi = \sin^{-1}\left(\cos\phi_P\cos\lambda'\cos\phi' + \sin\phi_P\sin\phi'\right) \tag{5.7}$$

$$\lambda = \lambda_o + \gamma \cos^{-1}\left[\frac{-\cos\phi_P \sin\phi' + \sin\phi_P \cos\lambda' \cos\phi'}{\cos\phi}\right] \qquad (5.8)$$

where from global symmetry $\gamma = 1$ if $0 \le \lambda' \le \pi$ and $\gamma = -1$ if $-\pi \le \lambda' < 0$ .

### 4.2 Wind field

Because the horizontal wind is a vector quantity, transformations between the two grids require the wind components to be resolved along the new axes. The horizontal wind vector may be expressed in terms of either coordinate system as follows:

$$\underline{u} = u\underline{\hat{\lambda}} + v\underline{\hat{\phi}} = u'\underline{\hat{\lambda}}' + v'\underline{\hat{\phi}}' \qquad (5.9)$$

On taking the dot product of (5.9) with each of the unit vectors, the following relationships between the 'zonal' and 'meridional' winds in each coordinate system are obtained:

$$u' = u\underline{\hat{\lambda}}\cdot\underline{\hat{\lambda}}' + v\underline{\hat{\phi}}\cdot\underline{\hat{\lambda}}' \text{ and } v' = u\underline{\hat{\lambda}}\cdot\underline{\hat{\phi}}' + v\underline{\hat{\phi}}\cdot\underline{\hat{\phi}}' \qquad (5.10)$$

$$u = u'\underline{\hat{\lambda}}\cdot\underline{\hat{\lambda}}' + v'\underline{\hat{\lambda}}\cdot\underline{\hat{\phi}}' \text{ and } v = u'\underline{\hat{\phi}}\cdot\underline{\hat{\lambda}}' + v'\underline{\hat{\phi}}\cdot\underline{\hat{\phi}}' \qquad (5.11)$$

These equations may be simplified somewhat since it follows from the dot product definition $\underline{a}\cdot\underline{b} = ab\cos\alpha$ , where $\alpha$ is the angle between $\underline{a}$ and $\underline{b}$, that

$$\underline{\hat{\lambda}}\cdot\underline{\hat{\lambda}}' = \underline{\hat{\phi}}\cdot\underline{\hat{\phi}}' \text{ and } \underline{\hat{\lambda}}'\cdot\underline{\hat{\phi}} = -\underline{\hat{\lambda}}\cdot\underline{\hat{\phi}}' \qquad (5.12)$$

Thus

$$u' = c_1 u - c_2 v \text{ and } v' = c_1 v + c_2 u \qquad (5.13)$$

$$u' = c_1 u' + c_2 v' \text{ and } v = c_1 v' - c_2 u' \qquad (5.14)$$

From the theory of general orthogonal curvilinear coordinates it may be shown that

$$\underline{\hat{\lambda}} = -\sin(\lambda-\lambda_0)\ \underline{i} + \cos(\lambda-\lambda_0)\ \underline{j} \tag{5.15}$$

$$\underline{\hat{\phi}} = -\sin\phi\ \cos(\lambda-\lambda_0)\ \underline{i} - \sin\phi\ \sin(\lambda-\lambda_0)\ \underline{j} + \cos\phi\ \underline{k} \tag{5.16}$$

$$\underline{\hat{\lambda}}' = -\sin\lambda'\ \sin\phi_p\ \underline{i} + \cos\lambda'\ \underline{j} - \sin\lambda'\ \cos\phi_p\ \underline{k} \tag{5.17}$$

$$\begin{aligned}\underline{\hat{\phi}}' = &- (\sin\phi'\ \cos\lambda'\ \sin\phi_p + \cos\phi'\ \cos\phi_p)\ \underline{i} - \sin\phi'\ \sin\lambda'\ \underline{j} \\ &- (\sin\phi'\ \cos\lambda'\ \cos\phi_p - \cos\phi'\ \sin\phi_p)\ \underline{k}\end{aligned} \tag{5.18}$$

Thus

$$c_1 = -\sin(\lambda-\lambda_0)\sin\lambda'\ \sin\phi_p + \cos(\lambda-\lambda_0)\cos\lambda' \tag{5.19}$$

and

$$\begin{aligned}c_2 = &\sin\phi\ \sin\lambda'\ \cos(\lambda-\lambda_0)\ \sin\phi_p - \sin\phi\ \cos\lambda'\ \sin(\lambda-\lambda_0) \\ &- \sin\lambda'\ \cos\phi\ \cos\phi_p\end{aligned} \tag{5.20}$$

In practice, since $c_1$ and $c_2$ are the sine and cosine of the local angle of rotation between the grids, it follows that

$$c_1^2 + c_2^2 = 1 \tag{5.21}$$

and a computationally economic evaluation of $c_2$ may be obtained directly from equations (5.19) and (5.21).

# REFERENCES

Dickinson,A. 1984 A possible new grid and integration area for the fine mesh model. Met O 11 Working Paper No 75.

Swinbank,R. and Wilson.C 1990 Vertical interpolation of temperature observations and model data. Short Range Forecasting Research Tech Note 48.

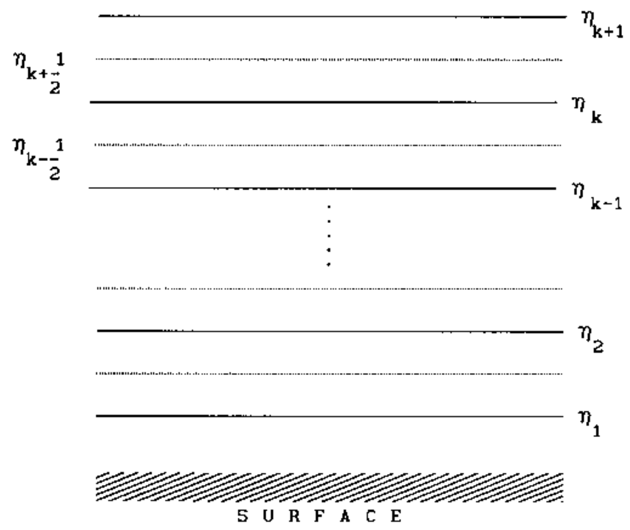

Figure 1. Distribution of layers in the hybrid coordinate system.

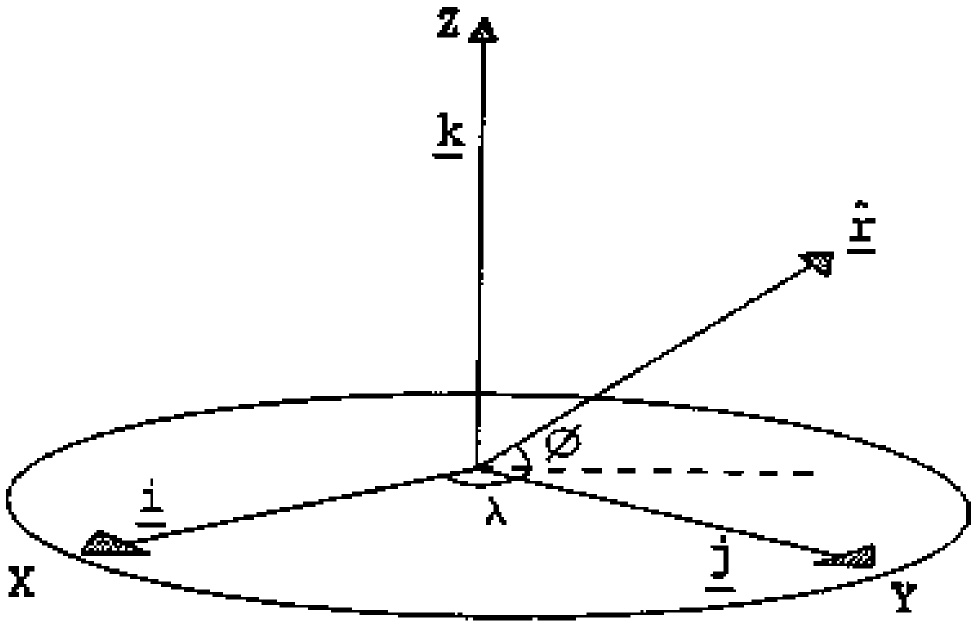

Figure 2. The equatorial plane of the standard latitude longitude grid.

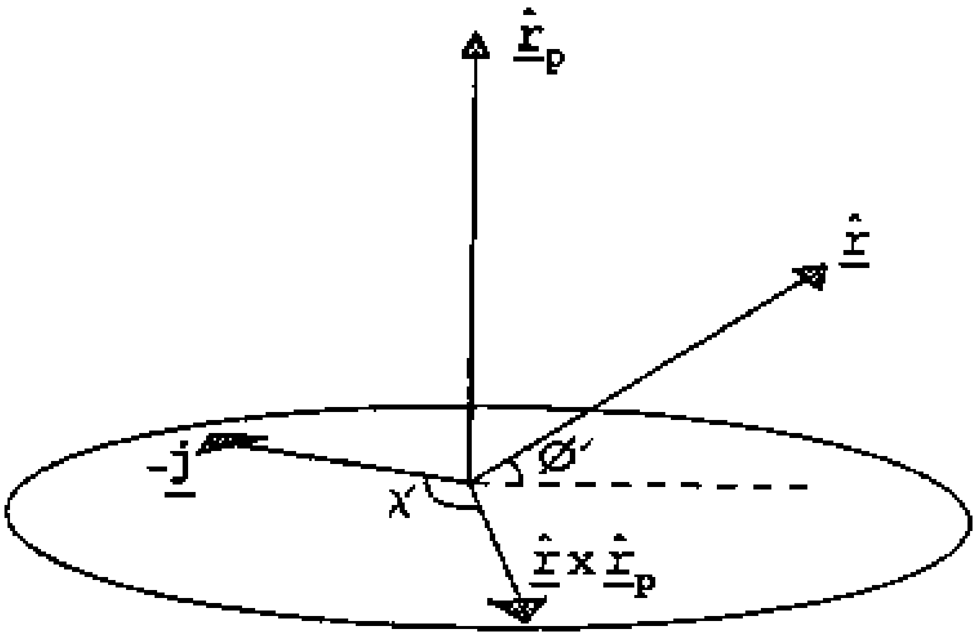

Figure 3. The equatorial plane of the transformed grid.

# APPENDIX 1

## INTERPOLATION SUBROUTINE CALLS

Librarian’s note: all revision information must appear at the front of the entire paper S1, which is now a single wordprocessor document.

## HORIZONTAL INTERPOLATION

### SUBROUTINE H_INT_INIT

*Description*

Interface to routines to initialise indices and weights for horizontal interpolation.

*Call*

```
      CALL H_INT_INIT
     &(ICOF,IDIM.P_FIELD_OUT,P_ROWS_IN,P_ROWS_OUT,
     & ROW_LENGTH_IN,ROW_LENGTH_OUT,U_FIELD_IN,U_FIELD_OUT,
     & U_ROWS_IN,U_ROWS_OUT,GLOBAL,GRIB,H_INT_TYPE,FIXHD_IN,FIXHD_OUT
     & REALHD_IN,REALHD_OUT,AW_AREA_BOX,
     & AW_INDEX_TARG_LHS,AW_INDEX_TARG_TOP,
     & BL_INDEX_B_L,BL_INDEX_B_R,BL_INDEX_NEAREST
     & AW_COLAT_T,AW_LONG_L,
     & COEFF1,COEFF2,COEFF3,COEFF4,
     & WEIGHT_B_L,WEIGHT_B_R,WEIGHT_T_L,WEIGHT_T_R)
```

*Arguments*

| | | | |
|---|---|---|---|
| ICOF | INTEGER | IN | Second dimension of coefficients array |
| IDIM | INTEGER | IN | Second dimension of index arrays |
| P_FIELD_OUT | INTEGER | IN | Total number of P-points on target grid |
| P_ROWS_IN | INTEGER | IN | Number of P-rows on source grid |
| P_ROWS_OUT | INTEGER | IN | Number of P-rows on target grid |
| ROW_LENGTH_IN | INTEGER | IN | Number of points per row on source grid |
| ROW_LENGTH_OUT | INTEGER | IN | Number of points per row on target grid |
| U_FIELD_IN | INTEGER | IN | Total number of U-points on source grid |
| U_FIELD_OUT | INTEGER | IN | Total number of U-points on target grid |
| U_ROWS_IN | INTEGER | IN | Number of U-rows on source grid |
| U_ROWS_OUT | INTEGER | IN | Number of U-rows on target grid |
| GLOBAL | LOGICAL | IN | True if global area required |
| GRIB | LOGICAL | IN | True if winds imported on A-grid |
| H_INT_TYPE | LOGICAL | IN | True = Area weighted interpolation<br>False = Bi-linear interpolation |
| FIXHD_IN(*) | INTEGER | IN | Fixed length header for source grid |
| FIXHD_OUT(*) | INTEGER | IN | Fixed length header for target grid |
| REALHD_IN(*) | REAL | IN | Real header for source grid |
| READHD_OUT(*) | REAL | IN | Real header for target grid |
| AW_AREA_BOX(IDIM) | REAL | IN | area of grid box in sq units of source grid |
| AW_INDEX_TARG_LHS ROW_LENGTH_OUT+1, IDIM) | INTEGER | IN | Index of source box overlapping lhs of target grid-box |
| AW_INDEX_TARG_TOP | INTEGER | IN | Index of source box overlapping |

|  |  |  |  |
|---|---|---|---|
| ROWS_OUT+1,IDIM) |  |  | top of target grid-box |
| BL_INDEX_B_L( LEN_FIELD_OUT, IDIM) | INTEGER | IN | Index of bottom lefthand corner of source gridbox |
| BL_INDEX_B_R( LEN_FIELD_OUT) | INTEGER | IN | Index of bottom righthand corner of source gridbox |
| BL_INDEX_NEAREST( IDIM) | INTEGER | IN | Gather index for nearest point on source grid for each target P-point |
| AW_COLAT_T ROWS_OUT+1, IDIM) | REAL | IN | Colatitide of top of target grid-box (in units of DELTA_LONG_SRCE) |
| AW_LONG_L ROW_LENGTH_OUT+1, IDIM) | REAL | IN | Left longitude of target grid-box (in units of DELTA_LONG_SRCE) |
| COEFF1(U_FIELD_OUT, ICOF) | REAL | IN | Coefficient of rotation no 1 on target grid |
| COEFF2(U_FIELD_OUT, ICOF) | REAL | IN | Coefficient of rotation no 2 on target grid |
| COEFF3(U_FIELD_IN, ICOF) | REAL | IN | Coefficient of rotation no 1 on source grid |
| COEFF4(U_FIELD_IN, ICOF) | REAL | IN | Coefficient of rotation no 2 on source grid |
| WEIGHT_T_R( LEN_FIELD_OUT, IDIM) | REAL | IN | Weight applied to value at top right hand corner of source gridbox |
| WEIGHT_B_L( LEN_FIELD_OUT, IDIM) | REAL | IN | Weight applied to value at bottom left hand corner of source gridbox |
| WEIGHT_B_R( LEN_FIELD_OUT, IDIM) | REAL | IN | Weight applied to value at bottom right hand corner of source gridbox |
| WEIGHT_T_L( LEN_FIELD_OUT, IDIM) | REAL | IN | Weight applied to value at top left hand corner of source gridbox |

### SUBROUTINE H_INT_INIT_AW

*Description*

Initialises indices and weights for area weighted horizontal interpolation.

*Call*

```
      CALL H_INT_INIT_AW
     &(ICOF,IDIM.P_FIELD_OUT,P_ROWS_IN,P_ROWS_OUT,
     & ROW_LENGTH_IN,ROW_LENGTH_OUT,U_FIELD_IN,U_FIELD_OUT,
     & U_ROWS_IN,U_ROWS_OUT,GLOBAL,GRIB,FIXHD_IN,FIXHD_OUT
     & REALHD_IN,REALHD_OUT,AW_AREA_BOX,
     & AW_INDEX_TARG_LHS,AW_INDEX_TARG_TOP,
     & BL_INDEX_B_L,BL_INDEX_B_R,BL_INDEX_NEAREST
     & AW_COLAT_T,AW_LONG_L,
     & WEIGHT_B_L,WEIGHT_B_R,WEIGHT_T_L,WEIGHT_T_R)
```

*Arguments*

|  |  |  |  |
|---|---|---|---|
| ICOF | INTEGER | IN | Second dimension of coefficients array |
| IDIM | INTEGER | IN | Second dimension of index arrays |
| P_FIELD_OUT | INTEGER | IN | Total number of P-points on target grid |

|  |  |  |  |
|---|---|---|---|
| P_ROWS_IN | INTEGER | IN | Number of P-rows on source grid |
| P_ROWS_OUT | INTEGER | IN | Number of P-rows on target grid |
| ROW_LENGTH_IN | INTEGER | IN | Number of points per row on source grid |
| ROW_LENGTH_OUT | INTEGER | IN | Number of points per row on target grid |
| U_FIELD_IN | INTEGER | IN | Total number of U-points on source grid |
| U_FIELD_OUT | INTEGER | IN | Total number of U-points on target grid |
| U_ROWS_IN | INTEGER | IN | Number of U-rows on source grid |
| U_ROWS_OUT | INTEGER | IN | Number of U-rows on target grid |
| GLOBAL | LOGICAL | IN | True if global area required |
| GRIB | LOGICAL | IN | True if  winds imported on A-grid |
| FIXHD_IN(*) | INTEGER | IN | Fixed length header for source grid |
| FIXHD_OUT(*) | INTEGER | IN | Fixed length header for target grid |
| REALHD_IN(*) | REAL | IN | Real header for source grid |
| READHD_OUT(*) | REAL | IN | Real header for target grid |
| AW_AREA_BOX(IDIM) | REAL | IN | area of grid box in sq units of source grid |
| AW_INDEX_TARG_LHS ROW_LENGTH_OUT+1, IDIM) | INTEGER | IN | Index of source box overlapping lhs of target grid-box |
| AW_INDEX_TARG_TOP ROWS_OUT+1,IDIM) | INTEGER | IN | Index of source box overlapping top of target grid-box |
| BL_INDEX_B_L( LEN_FIELD_OUT, IDIM) | INTEGER | IN | Index of bottom lefthand corner of source gridbox |
| BL_INDEX_B_R( LEN_FIELD_OUT) | INTEGER | IN | Index of bottom righthand corner of source gridbox |
| BL_INDEX_NEAREST( IDIM) | INTEGER | IN | Gather index for nearest point on source grid for each target P-point |
| AW_COLAT_T ROWS_OUT+1, IDIM) | REAL | IN | Colatitide of top of target grid-box (in units of DELTA_LONG_SRCE) |
| AW_LONG_L ROW_LENGTH_OUT+1, IDIM) | REAL | IN | Left longitude of  target grid-box (in units of DELTA_LONG_SRCE) |
| WEIGHT_T_R( LEN_FIELD_OUT, IDIM) | REAL | IN | Weight applied to value at top right hand corner of source gridbox |
| WEIGHT_B_L( LEN_FIELD_OUT, IDIM) | REAL | IN | Weight applied to value at bottom left hand corner of source gridbox |
| WEIGHT_B_R( LEN_FIELD_OUT, IDIM) | REAL | IN | Weight applied to value at bottom right hand corner of source gridbox |
| WEIGHT_T_L( LEN_FIELD_OUT, IDIM) | REAL | IN | Weight applied to value at top left hand corner of source gridbox |

## SUBROUTINE H_INT_INIT_BL

*Description*

Initialises indices and weights for bilinear horizontal interpolation.

*Call*

```
      CALL H_INT_INIT_BL
     &(ICOF,IDIM.P_FIELD_OUT,P_ROWS_IN,P_ROWS_OUT,
     & ROW_LENGTH_IN,ROW_LENGTH_OUT,U_FIELD_IN,U_FIELD_OUT,
     & U_ROWS_IN,U_ROWS_OUT,GLOBAL,GRIB,FIXHD_IN,FIXHD_OUT
```

```
     & REALHD_IN,REALHD_OUT,BL_INDEX_B_L,BL_INDEX_B_R,BL_INDEX_NEAREST,
     & WEIGHT_B_L,WEIGHT_B_R,WEIGHT_T_L,WEIGHT_T_R)
```

*Arguments*

| | | | |
|---|---|---|---|
| ICOF | INTEGER | IN | Second dimension of coefficients array |
| IDIM | INTEGER | IN | Second dimension of index arrays |
| P_FIELD_OUT | INTEGER | IN | Total number of P-points on target grid |
| P_ROWS_IN | INTEGER | IN | Number of P-rows on source grid |
| P_ROWS_OUT | INTEGER | IN | Number of P-rows on target grid |
| ROW_LENGTH_IN | INTEGER | IN | Number of points per row on source grid |
| ROW_LENGTH_OUT | INTEGER | IN | Number of points per row on target grid |
| U_FIELD_IN | INTEGER | IN | Total number of U-points on source grid |
| U_FIELD_OUT | INTEGER | IN | Total number of U-points on target grid |
| U_ROWS_IN | INTEGER | IN | Number of U-rows on source grid |
| U_ROWS_OUT | INTEGER | IN | Number of U-rows on target grid |
| GLOBAL | LOGICAL | IN | True if global area required |
| GRIB | LOGICAL | IN | True if  winds imported on A-grid |
| FIXHD_IN(*) | INTEGER | IN | Fixed length header for source grid |
| FIXHD_OUT(*) | INTEGER | IN | Fixed length header for target grid |
| REALHD_IN(*) | REAL | IN | Real header for source grid |
| READHD_OUT(*) | REAL | IN | Real header for target grid |
| INDEX_B_L( LEN_FIELD_OUT, IDIM) | INTEGER | IN | Index of bottom lefthand corner of source gridbox |
| INDEX_B_R( LEN_FIELD_OUT) | INTEGER | IN | Index of bottom righthand corner of source gridbox |
| INDEX_NEAREST( IDIM) | INTEGER | IN | Gather index for nearest point on source grid for each target P-point |
| WEIGHT_T_R( LEN_FIELD_OUT, IDIM) | REAL | IN | Weight applied to value at top right hand corner of source gridbox |
| WEIGHT_B_L( LEN_FIELD_OUT, IDIM) | REAL | IN | Weight applied to value at bottom left hand corner of source gridbox |
| WEIGHT_B_R( LEN_FIELD_OUT, IDIM) | REAL | IN | Weight applied to value at bottom right hand corner of source gridbox |
| WEIGHT_T_L( LEN_FIELD_OUT, IDIM) | REAL | IN | Weight applied to value at top left hand corner of source gridbox |

### SUBROUTINE H_INT_CTL

*Description*

Interface to horizontal interpolation routines H_INT_AW and H_INT_BL

*Call*

```
      CALL H_INT_CTL
     &(IDIM.LEN_FIELD_OUT,ROW_LENGTH_IN,ROW_LENGTH_OUT,ROWS_IN,ROWS_OUT,
     & AW_AREA_BOX,GLOBAL,H_INT_TYPE,
     & AW_INDEX_TARG_LHS,AW_INDEX_TARG_TOP,BL_INDEX_B_L,BL_INDEX_B_R,
     & AW_COLAT_T,AW_LONG_L,DATA_IN,
     & WEIGHT_B_L,WEIGHT_B_R,WEIGHT_T_L,WEIGHT_T_R,DATA_OUT)
```

*Arguments*

| | | | |
|---|---|---|---|
| IDIM | INTEGER | IN | Now redundant |
| LEN_FIELD_OUT | INTEGER | IN | Total number of points on target grid |
| ROWS_IN | INTEGER | IN | Number of rows on source grid |
| ROWS_OUT | INTEGER | IN | Number of rows on target grid |
| ROW_LENGTH_IN | INTEGER | IN | Number of points per row on source grid |
| ROW_LENGTH_OUT | INTEGER | IN | Number of points per row on target grid |
| AW_AREA_BOX | REAL | IN | area of grid box in sq units of source grid |
| GLOBAL | LOGICAL | IN | True if global area required |
| H_INT_TYPE | LOGICAL | IN | True = Area weighted interpolation<br>False = Bi-linear interpolation |
| AW_INDEX_TARG_LHS ROW_LENGTH_OUT+1) | INTEGER | IN | Index of source box overlapping lhs of target grid-box |
| AW_INDEX_TARG_TOP ROWS_OUT+1) | INTEGER | IN | Index of source box overlapping top of target grid-box |
| BL_INDEX_B_L(LEN_FIELD_OUT) | INTEGER | IN | Index of bottom lefthand corner of source gridbox |
| BL_INDEX_B_R(LEN_FIELD_OUT) | INTEGER | IN | Index of bottom righthand corner of source gridbox |
| AW_COLAT_T ROWS_OUT+1) | INTEGER | IN | Colatitide of top of target grid-box (in units of DELTA_LONG_SRCE) |
| AW_LONG_L ROW_LENGTH_OUT+1 | INTEGER | IN | Left longitude of target grid-box (in units of DELTA_LONG_SRCE) |
| WEIGHT_T_R(LEN_FIELD_OUT) | REAL | IN | Weight applied to value at top right hand corner of source gridbox |
| WEIGHT_B_L(LEN_FIELD_OUT) | REAL | IN | Weight applied to value at bottom left hand corner of source gridbox |
| WEIGHT_B_R(LEN_FIELD_OUT) | REAL | IN | Weight applied to value at bottom right hand corner of source gridbox |
| WEIGHT_T_L(LEN_FIELD_OUT) | REAL | IN | Weight applied to value at top left hand corner of source gridbox |
| DATA_IN(ROWS_IN*ROW_LENGTH_IN) | REAL | IN | Data to be interpolated on source grid |
| DATA_OUT(ROWS_OUT*ROW_LENGTH_OUT)) | REAL | OUT | Interpolated data on target grid |

### SUBROUTINE H_INT_AW

*Description*

Carries out area weighted horizontal interpolation using the coefficients and indices calculated in subroutine BOX_BND.

*Call*

```
      CALL H_INT_AW
     &(ROWS_IN,ROW_LENGTH_OUT,ROW_INDEX_TARG_LHS,ROW_LENGTH_OUT,GLOBAL,
     & AW_INDEX_TARG_LHS,AW_INDEX_TARG_TOP,
     & AW_COLAT_T,AW_LONG_L,DATA_IN,DATA_OUT)
```

*Arguments*

| | | | |
|---|---|---|---|
| ROWS_IN | INTEGER | IN | Number of rows on source grid |
| ROWS_OUT | INTEGER | IN | Number of rows on target grid |

| | | | |
|---|---|---|---|
| ROW_LENGTH_IN | INTEGER | IN | Number of points per row on source grid |
| ROW_LENGTH_OUT | INTEGER | IN | Number of points per row on target grid |
| GLOBAL | LOGICAL | IN | True if global area required |
| AW_INDEX_TARG_LHS ROW_LENGTH_OUT+1) | INTEGER | IN | Index of source box overlapping lhs of target grid-box |
| AW_INDEX_TARG_TOP ROWS_OUT+1) | INTEGER | IN | Index of source box overlapping top of target grid-box |
| AW_COLAT_T ROWS_OUT+1) | INTEGER | IN | Colatitide of top of target grid-box (in units of DELTA_LONG_SRCE) |
| AW_LONG_L ROW_LENGTH_OUT+1 | INTEGER | IN | Left longitude of target grid-box (in units of DELTA_LONG_SRCE) |
| DATA_IN( ROWS_IN* ROW_LENGTH_IN) | REAL | IN | Data to be interpolated on source grid |
| DATA_OUT( ROWS_OUT* ROW_LENGTH_OUT)) | REAL | OUT | Interpolated data on target grid |

SUBROUTINE H_INT_BL

*Description*

Carries out bi-linear horizontal interpolation using the coefficients and gather indices calculated in subroutine H_INT_CO.

*Call*

```
      CALL H_INT_BL
     &(ROWS_IN,ROW_LENGTH_IN,LEN_FIELD_OUT,INDEX_B_L,INDEX_B_R,DATA_IN
     & WEIGHT_B_L,WEIGHT_B_R,WEIGHT_T_L,WEIGHT_T_R,DATA_OUT)
```

*Arguments*

| | | | |
|---|---|---|---|
| ROWS_IN | INTEGER | IN | Number of rows on source grid |
| ROW_LENGTH_IN | INTEGER | IN | Number of points per row on source grid |
| LEN_FIELD_OUT | INTEGER | IN | Total number of points on target grid |
| INDEX_B_L( LEN_FIELD_OUT) | INTEGER | IN | Index of bottom lefthand corner of source gridbox |
| INDEX_B_R( LEN_FIELD_OUT) | INTEGER | IN | Index of bottom righthand corner of source gridbox |
| DATA_IN( ROWS_IN, ROW_LENGTH_IN) | REAL | IN | Data to be interpolated on source grid |
| WEIGHT_T_R( LEN_FIELD_OUT) | REAL | IN | Weight applied to value at top right hand corner of source gridbox |
| WEIGHT_B_L( LEN_FIELD_OUT) | REAL | IN | Weight applied to value at bottom left hand corner of source gridbox |
| WEIGHT_B_R( LEN_FIELD_OUT) | REAL | IN | Weight applied to value at bottom right hand corner of source gridbox |
| WEIGHT_T_L( LEN_FIELD_OUT) | REAL | IN | Weight applied to value at top left hand corner of source gridbox |
| DATA_OUT( LEN_FIELD_OUT) | REAL | OUT | Interpolated data on target grid |

SUBROUTINE H_INT_CO

*Description*

Calculates bi-linear horizontal interpolation coefficients and gather indices for interpolating between generalised latitude-longitude grids (eg non-uniform ocean grid or global, regional or rotated latitude-longitude grids). These fields are used as input to subroutines H_INT, NEAR_PT and COAST_AJ. The gather indices point to the bottom left hand and bottom right hand corners of each grid box on the source grid enclosing a target point. Two indices are ended to cater for east-west (latitudinal direction) cyclic boundaries when the source data is global or hemispheric. If the target point falls outside the area covered by the input grid, one-sided differencing is used.

*Call*

```
      CALL H_INT_CO
     *(INDEX_B_L,INDEX_B_R,WEIGHT_T_R,WEIGHT_B_R,WEIGHT_T_L,WEIGHT_B_L
     *,LAMBDA_SRCE,PHI_SRCE,LAMBDA_TARG,PHI_TARG
     *,POINTS_LAMBDA_SRCE,POINTS_PHI_SRCE,POINTS,CYCLIC)
```

*Arguments*

| | | | |
|---|---|---|---|
| POINTS_LAMBDA_SRCE | INTEGER | IN | Number of longitude points on source grid |
| POINTS_PHI_SRCE | INTEGER | IN | Number of latitude points on source grid |
| POINTS | INTEGER | IN | Total number of points on target grid |
| INDEX_B_L(POINTS) | INTEGER | OUT | Index of bottom lefthand corner of source gridbox |
| INDEX_B_R(POINTS) | INTEGER | OUT | Index of bottom righthand corner of source gridbox |
| LAMBDA_TARG(POINTS) | REAL | IN | Longitude coords of each point on target grid<br>in degrees using same rotation as source grid |
| PHI_TARG(POINTS) | REAL | IN | Latitude coords of each point on target grid in degrees using same rotation as source grid |
| WEIGHT_T_R(POINTS) | REAL | OUT | Weight applied to value at top right hand corner of source gridbox |
| WEIGHT_B_L(POINTS) | REAL | OUT | Weight applied to value at bottom left hand corner of source gridbox |
| WEIGHT_B_R(POINTS) | REAL | OUT | Weight applied to value at bottom right hand corner of source gridbox |
| WEIGHT_T_L(POINTS) | REAL | OUT | Weight applied to value at top left hand corner of source gridbox |
| LAMBDA_SRCE( POINTS_LAMBDA_SRCE) | REAL | IN | Longitude coordinates of source grid in degrees |
| PHI_SRCE( POINTS_PHI_SRCE) | REAL | IN | Latitude coords of source grid in degrees |
| CYCLIC | LOGICAL | IN | =T, then source data cyclic<br>=F, then source data non-cyclic |

SUBROUTINE BOX_BND

*Description*

Sets up longitude, colatitude and indexes of the overlapping source grid-boxes at left hand side and top of target grid-boxes for use in area weighted interpolation.

*Call*

```
      CALL BOX_BND
```

```
     &(I_L,LONG_L,J_T,COLAT_T,AREA_BOX,
     & ROW_LENGTH,ROWS,ROW_LENGTH_SRCE,ROWS_SRCE,
     & DELTA_LONG,DELTA_LAT,START_LONG,START_LAT,
     & DELTA_LONG_SRCE,DELTA_LAT_SRCE,START_LONG_SRCE,START_LAT_SRCE,
     & IGRID,IGRID_SRCE,GLOBAL)
```

*Arguments*

| | | | |
|---|---|---|---|
| I_L(ROW_LENGTH+1) | INTEGER | OUT | Index of source box overlapping lhs of target grid-box. |
| LONG_L(ROW_LENGTH+1) | REAL | OUT | Left longitude of target grid-box. (in units of DELTA_LONG_SRCE) |
| J_T(ROWS+1) | INTEGER | OUT | Index of source box overlapping lhs of target grid-box. |
| COLAT_T(ROWS+1) | REAL | OUT | Colatitude of top of target grid-box. (in units of DELTA_LONG_SRCE) |
| AREA_BOX | REAL | OUT | area of grid box in sq units of source. |
| ROW_LENGTH | INTEGER | IN | Number of points per row on target area. |
| ROWS | INTEGER | IN | Number of rows on target area. |
| ROW_LENGTH_SRCE | INTEGER | IN | Number of points per row on source area. |
| ROWS_SRCE | INTEGER | IN | Number of rows on source area. |
| DELTA_LONG | REAL | IN | Longitude increment of target grid (deg). |
| DELTA_LAT | REAL | IN | Latitude increment of target grid (deg). |
| START_LONG | REAL | IN | Start longitude of centre of first grid-box in target area. |
| START_LAT | REAL | IN | Start latitude of centre of first grid-box in target area. |
| DELTA_LONG_SRCE | REAL | IN | Longitude increment of source grid. |
| DELTA_LAT_SRCE | REAL | IN | Latitude increment of source grid. |
| START_LONG_SRCE | REAL | IN | Start longitude of centre of first grid-box in source area. |
| START_LAT_SRCE | REAL | IN | Start latitude of centre of first grid-box in source area. |
| IGRID | INTEGER | IN | Grid indicator 1=p-grid,2=u-grid (target) |
| IGRID_SRCE | INTEGER | IN | Grid indicator 1=p-grid,2=u-grid (source) |
| GLOBAL | LOGICAL | IN | True if global area required. |

### SUBROUTINE BOX_SUM

*Description*

Sums contributions from gridboxes for source data on a regular lat-long grid to form means for gridboxes of a regular lat-long grid specified as target.
Used in area weighted interpolation.

*Call*

```
      CALL BOX_SUM
     &(SOURCE_ROW_LENGTH,SOURCE_ROWS,ROW_LENGTH,ROWS,
     & LONG_L,COLAT_T,I_L,J_T,GLOBAL,BOXSUM,SOURCE)
```

*Arguments*

| | | | |
|---|---|---|---|
| SOURCE_ROW_LENGTH | INTEGER | IN | Number of points per row on source area. |
| SOURCE_ROWS | INTEGER | IN | Number of rows on source area. |
| ROW_LENGTH | INTEGER | IN | Number of points per row on target area. |

| | | | |
|---|---|---|---|
| ROWS | INTEGER | IN | Number of rows on target area. |
| LONG_L(ROW_LENGTH+1) | REAL | IN | Left longitude of target grid-box. (in units of DELTA_LONG_SRCE) |
| COLAT_T(ROWS+1) | REAL | IN | Colatitude of top of target grid-box. (in units of DELTA_LONG_SRCE) |
| I_L(ROW_LENGTH+1) | INTEGER | IN | Index of source box overlapping lhs of target grid-box. |
| J_T(ROWS+1) | INTEGER | IN | Index of source box overlapping lhs of target grid-box. |
| GLOBAL | LOGICAL | IN | True if global area required. |
| BOXSUM( ROW_LENGTH,ROWS) | REAL | OUT | Sum of data on target grid. |
| SOURCE( SOURCE_ROW_LENGTH, SOURCE_ROWS) | REAL | IN | source data. |

## SUBROUTINE COAST_AJ

*Description*

(i) Produces gather indices which map each coastal point on the target grid onto its nearest point on the source grid. This allows correction of those surface fields which are non-homogeneous across land/sea boundaries after horizontal interpolation by subroutine H_INT. The algorithm uses linear interpolation weights and gather indices calculated by H_INT_CO.

(ii) If a land-sea mask for the target grid is not provided, one is created. When a target land/sea mask is provided, a further index is output containing those points on the target grid for which the 4 surrounding source points are not of the same land/sea type as the target point. These points will generally point to new islands etc that are resolved by a high resolution land/sea mask. These should be set to appropriate values (eg climatology) as required.

*Call*

```
      CALL COAST_AJ
     *(INDEX_B_L,INDEX_B_R,WEIGHT_T_R,WEIGHT_B_R,WEIGHT_T_L,WEIGHT_B_L
     *,POINTS_LAMBDA_SRCE,POINTS_PHI_SRCE,POINTS,LAND_SEA_SRCE
     *,LAND_SEA_TARG,INDEX_TARG,INDEX_SRCE,COASTAL_POINTS,MASK
     *,INDEX_TARG_SEA_UNRES,SEA_POINTS_UNRES
     *,INDEX_TARG_LAND_UNRES,LAND_POINTS_UNRES)
```

*Arguments*

| | | | |
|---|---|---|---|
| POINTS_LAMBDA_SRCE | INTEGER | IN | Number of lambda points on source grid |
| POINTS_PHI_SRCE | INTEGER | IN | Number of phi points on source grid |
| POINTS | INTEGER | IN | Total number of points on target grid |
| COASTAL_POINTS | INTEGER | OUT | Number of coastal points on target grid |
| SEA_POINTS_UNRES | INTEGER | OUT | No. of unresolved sea points when MASK=T |
| LAND_POINTS_UNRES | INTEGER | OUT | No. of unresolved land points when MASK=T |
| INDEX_B_L(POINTS) | INTEGER | IN | Index of bottom lefthand corner of source gridbox |
| INDEX_B_R(POINTS) | INTEGER | IN | Index of bottom righthand corner of source gridbox |
| LAND_SEA_TARG(POINTS) | INTEGER | OUT<br>IN | Land/sea mask on target grid.<br>If MASK=T then precalculated land/sea mask |
| LAND_SEA_SRCE(POINTS_LAMBDA_SRCE*POINTS_PHI_SRCE) | | | |

|  |  |  |  |
|---|---|---|---|
|  | INTEGER | IN | Land/sea mask on source grid |
| INDEX_TARG(POINTS) | INTEGER | OUT | Index of target coastal points |
| INDEX_SRCE(POINTS) | INTEGER | OUT | Index of source points mapped onto target coastal points |
| INDEX_TARG_SEA_UNRES(POINTS) | INTEGER | OUT | Index of sea points on target grid which are unresolved. |
| INDEX_TARG_LAND_UNRES(POINTS) | INTEGER | OUT | Index of land points on target grid which are unresolved. |
| WEIGHT_T_R(POINTS) | REAL | IN | Weight applied to value at top right hand corner of source gridbox |
| WEIGHT_B_L(POINTS) | REAL | IN | Weight applied to value at bottom left hand corner of source gridbox |
| WEIGHT_B_R(POINTS) | REAL | IN | Weight applied to value at bottom right hand corner of source gridbox |
| WEIGHT_T_L(POINTS) | REAL | IN | Weight applied to value at top left hand corner of source gridbox |
| MASK | LOGICAL | IN | =F, then l/s mask estimated<br>=T, then l/s mask input as LAND_SEA_TARG |

SUBROUTINE INTF_COAST_AJ

*Description*

When using the SPIRAL_S routine to set values at unresolved points. This routine acts as an interface between subroutines CONTROL and SPIRAL_S. It calculates the smallest necessary search radius NSEARCH and then calls SPIRAL_S.

*Call*

```
      CALL INTF_COAST_AJ
     *(LAND_SEA_MASK,INDEX_UNRES,NO_POINT_UNRES,POINTS_PHI,POINTS_LAMBDA
     *,DATA_FIELD,SEA_LAND,CYCLIC,MAXDIM)
```

*Arguments*

|  |  |  |  |
|---|---|---|---|
| LAND_SEA_MASK(POINTS_LAMBDA,POINTS_PHI) | INTEGER | IN | Land-sea mask on target grid |
| INDEX_UNRES(POINTS_LAMBDA,POINTS_PHI) | INTEGER | IN | Index to unresolved points |
| NO_POINTS_UNRES | INTEGER | IN | Number of unresolved points |
| POINTS_PHI | INTEGER | IN | Number of rows on target grid |
| POINTS_LAMBDA | INTEGER | IN | Number of columns on target grid |
| DATA_FIELD | REAL | IN<br>OUT | Data to be corrected<br>Corrected data |
| SEA_LAND | INTEGER | IN | =0 for sea field =1/-1 for land field |
| CYCLIC | LOGICAL | IN | =T if data covers complete latitude cyclic |
| MAXDIM | INTEGER | IN | Largest dimension of field |

## SUBROUTINE SPIRAL_S

*Description*

Attempts to set a value at points which are unresolved when interpolating between one grid and another. A value is set by finding the mean of surrounding points which do have data set within a search radius determined by NSEARCH.

*Call*

```
      CALL SPIRAL_S
     *(LAND_SEA_MASK,INDEX_UNRES,NO_POINT_UNRES,POINTS_PHI,POINTS_LAMBDA
     *,DATA_FIELD,NSEARCH,SEA_LAND,CYCLIC)
```

*Arguments*

| | | | |
|---|---|---|---|
| LAND_SEA_MASK(POINTS_LAMBDA,POINTS_PHI) | INTEGER | IN | Land-sea mask on target grid |
| INDEX_UNRES(POINTS_LAMBDA,POINTS_PHI) | INTEGER | IN | Index to unresolved points |
| NO_POINTS_UNRES | INTEGER | IN | Number of unresolved points |
| POINTS_PHI | INTEGER | IN | Number of rows on target grid |
| POINTS_LAMBDA | INTEGER | IN | Number of columns on target grid |
| DATA_FIELD | REAL | IN | Data to be corrected |
| | | OUT | Corrected data |
| NSEARCH | INTEGER | IN | Number of points in each direction to search |
| SEA_LAND | INTEGER | IN | =0 for sea field =1/-1 for land field |
| CYCLIC | LOGICAL | IN | =T if data covers complete latitude cyclic |

## SUBROUTINE NEAR_PT

*Description*

Produces gather indices which map each point on the target grid onto its nearest point on the source grid. This allows horizontal interpolation by choosing the value of the nearest neighbour.

*Call*

```
      CALL NEAR_PT(INDEX_BL,INDEX_B_R,WEIGHT_T_R,WEIGHT_B_R,WEIGHT_T_L,
     *,WEIGHT_B_L,POINTS,POINTS_LAMBDA_SRCE,INDEX_NEAREST)
```

*Arguments*

| | | | |
|---|---|---|---|
| POINTS_LAMBDA_SRCE | INTEGER | IN | Number of lambda points on source grid |
| POINTS | INTEGER | IN | Total number of points on target grid |
| INDEX_NEAREST(POINTS) | INTEGER | OUT | Index of nearest source point to each target point. |
| INDEX_B_L(POINTS) | INTEGER | IN | Index of bottom lefthand corner of source gridbox |
| INDEX_B_R(POINTS) | INTEGER | IN | Index of bottom righthand corner of source gridbox |
| WEIGHT_T_R(POINTS) | REAL | IN | Weight applied to value at top right hand corner of source gridbox |
| WEIGHT_B_L(POINTS) | REAL | IN | Weight applied to value at bottom left hand corner of source gridbox |
| WEIGHT_B_R(POINTS) | REAL | IN | Weight applied to value at bottom right hand corner of source gridbox |

|  |  |  |  |
|---|---|---|---|
| WEIGHT_T_L(POINTS) | REAL | IN | Weight applied to value at top left hand corner of source gridbox |

### SUBROUTINE P_TO_UV

*Description*

Interpolates a horizontal field from pressure to wind points on an Arakawa B grid. Under UPDATE identifier GLOBAL the data is assumed periodic along rows. Otherwise, the first and last value on each row are calculated using one-sided differencing. The output array contains one less row than the input array.

*Call*

```
      CALL P_TO_UV(P_DATA,U_DATA,P_FIELD,U_FIELD,ROW_LENGTH,ROWS)
```

*Arguments*

|  |  |  |  |
|---|---|---|---|
| ROWS | INTEGER | IN | Number of rows to be updated. |
| ROW_LENGTH | INTEGER | IN | Number of points per row |
| P_FIELD | INTEGER | IN | Number of points in input field |
| U_FIELD | INTEGER | IN | Number of points in output field |
| P_DATA(P_FIELD) | REAL | INOUT | Data on p points |
| U_DATA(U_FIELD) | REAL | OUT | Data on uv points |

### SUBROUTINE UV_TO_P

*Description*

Interpolates a horizontal field from wind to pressure points on an Arakawa B grid. Under UPDATE identifier GLOBAL the data is assumed periodic along rows. Otherwise, the first and last value on each row is calculated using one-sided differencing. The output array contains one less row than the input array.

*Call*

```
      CALL UV_TO_P(U_DATA,P_DATA,U_FIELD,P_FIELD,ROW_LENGTH,ROWS)
```

*Arguments*

|  |  |  |  |
|---|---|---|---|
| ROWS | INTEGER | IN | Number of rows to be updated. |
| ROW_LENGTH | INTEGER | IN | Number of points per row |
| P_FIELD | INTEGER | IN | Number of points in output field |
| U_FIELD | INTEGER | IN | Number of points in input field |
| P_DATA(P_FIELD) | REAL | INOUT | Data on p points |
| U_DATA(U_FIELD) | REAL | OUT | Data on uv points |

## GRID TRANSFORMATIONS

### SUBROUTINE EQTOLL

*Description*

Calculates latitude and longitude on standard grid from input arrays of latitude and longitude on equatorial latitude-longitude grid used in regional models. Both input and output latitudes and longitudes are in degrees. Output latitudes are in the range 0 – 360 .

*Call*

```
      CALL EQTOLL(PHI_EQ,LAMBDA_EQ,PHI,LAMBDA,PHI_POLE,LAMBDA_POLE,POINTS)
```

*Arguments*

| | | | |
|---|---|---|---|
| POINTS | INTEGER | IN | Number of points to be processed |
| PHI(POINTS) | REAL | OUT | Latitude |
| LAMBDA(POINTS) | REAL | OUT | Longitude |
| LAMBDA_EQ(POINTS) | REAL | IN | Longitude in equatorial lat-lon coords |
| PHI_EQ(POINTS) | REAL | IN | Latitude in equatorial lat-lon coords |
| PHI_POLE | REAL | IN | Latitude of equatorial lat-lon pole |
| LAMBDA_POLE | REAL | IN | Longitude of equatorial lat-lon pole |

### SUBROUTINE LLTOEQ

*Description*

Calculates latitude and longitude on equatorial latitude-longitude grid used in regional models from input arrays of latitude and longitude on standard grid. Both input and output latitudes and longitudes are in degrees. Output latitudes are in the range 0 – 360 .

*Call*

```
      CALL
LLTOEQ(PHI,LAMBDA,PHI_EQ,LAMBDA_EQ,PHI_POLE,LAMBDA_POLE,POINTS,POINTS2)
```

*Arguments*

| | | | |
|---|---|---|---|
| POINTS | INTEGER | IN | Number of u-points to be processed |
| POINTS2 | INTEGER | IN | Number of v-points to be processed |
| PHI(POINTS) | REAL | IN | Latitude |
| LAMBDA(POINTS) | REAL | IN | Longitude |
| LAMBDA_EQ(POINTS) | REAL | OUT | Longitude in equatorial lat-lon coords |
| PHI_EQ(POINTS) | REAL | OUT | Latitude in equatorial lat-lon coords |
| PHI_POLE | REAL | IN | Latitude of equatorial lat-lon pole |
| LAMBDA_POLE | REAL | IN | Longitude of equatorial lat-lon pole |

### SUBROUTINE W_EQTOLL

*Description*

Calculates u and v components of wind on standard latitude-longitude grid by rotating wind components on equatorial latitude-longitude grid.

*Call*

```
      CALL W_EQTOLL(COEFF1,COEFF2,U_EQ,V_EQ,U,V,POINTS)
```

*Arguments*

| | | | |
|---|---|---|---|
| POINTS | INTEGER | IN | Number of u-points to be processed |
| POINTS2 | INTEGER | IN | Number of v-points to be processed |

| | | | |
|---|---|---|---|
| COEFF1(POINTS) | REAL | IN | Coefficient of rotation no 1 |
| COEFF2(POINTS) | REAL | IN | Coefficient of rotation no 2 |
| U_EQ(POINTS) | REAL | IN | u component of wind on equatorial grid |
| V_EQ(POINTS) | REAL | IN | v component of wind on equatorial grid |
| U(POINTS) | REAL | OUT | u component of wind on lat-lon grid |
| V(POINTS) | REAL | OUT | v component of wind on lat-lon grid |

## SUBROUTINE W_LLTOEQ

*Description*

Calculates u and v components of wind on equatorial latitude longitude grid by rotating wind components on standard latitude-longitude grid.

*Call*

```
      CALL W_LLTOEQ(COEFF1,COEFF2,U,V,U_EQ,V_EQ,POINTS)
```

*Arguments*

| | | | |
|---|---|---|---|
| POINTS | INTEGER | IN | Number of points to be processed |
| COEFF1(POINTS) | REAL | IN | Coefficient of rotation no 1 |
| COEFF2(POINTS) | REAL | IN | Coefficient of rotation no 2 |
| U_EQ(POINTS) | REAL | OUT | u component of wind on equatorial grid |
| V_EQ(POINTS) | REAL | OUT | v component of wind on equatorial grid |
| U(POINTS) | REAL | IN | u component of wind on lat-lon grid |
| V(POINTS) | REAL | IN | v component of wind on lat-lon grid |

## SUBROUTINE W_COEFF

*Description*

Calculates coefficients used to translate u and v components of wind between equatorial latitude-longitude grid and standard latitude-longitude grid (or visa versa). Output latitudes and longitudes are in degrees.

*Call*

```
      CALL W_COEFF(COEFF1,COEFF2,LAMBDA,LAMBDA_EQ,PHI_POLE,LAMBDA_POLE
     *,POINTS)
```

*Arguments*

| | | | |
|---|---|---|---|
| POINTS | INTEGER | IN | Number of points to be processed |
| COEFF1(POINTS) | REAL | OUT | Coefficient of rotation no 1 |
| COEFF2(POINTS) | REAL | OUT | Coefficient of rotation no 2 |
| LAMBDA(POINTS) | REAL | IN | Longitude |
| LAMBDA_EQ(POINTS) | REAL | IN | Longitude in equatorial lat-lon coords |
| PHI_POLE | REAL | IN | Latitude of equatorial lat-lon pole |
| LAMBDA_POLE | REAL | IN | Longitude of equatorial pole |

# VERTICAL INTERPOLATION

## SUBROUTINE V_INT_ZH

*Description*

Calculates the height of each layer boundary (half level) using the hydrostatic approximation (see equation (3.5)).

*Call*

```
      CALL
V_INT_ZH(P_EXNER_HALF,THETA,Q,PHI_STAR,ZH,POINTS,P_LEVELS,Q_LEVELS)
```

*Arguments*

| | | | |
|---|---|---|---|
| POINTS | INTEGER | IN | Number of horizontal points to be processed |
| P_LEVELS | INTEGER | IN | Number of model levels |
| Q_LEVELS | INTEGER | IN | Number of wet levels |
| P_EXNER_HALF (POINTS,P_LEVELS+1) | REAL | IN | Exner pressure at model half levels |
| THETA(POINTS,P_LEVELS) | REAL | IN | Potential temperature at full levels |
| Q(POINTS,Q_LEVELS) | REAL | IN | Specific humidity at full levels |
| PHI_STAR(POINTS) | REAL | IN | Geopotential height of topography |
| ZH(POINTS,P_LEVELS+1) | REAL | OUT | Height of model half levels |

## SUBROUTINE V_INT_Z

*Description*

Calculates the height of an arbitrary pressure surface using the technique described in section 3.2. The top model level is currently ignored when doing the calculations.

*Call*

```
      CALL V_INT_Z(P,PL,PSTAR,P_EXNER_HALF,THETA,Q,ZH,Z,POINTS
     * ,P_LEVELS_MODEL,Q_LEVELS_MODEL,L,AKH,BKH,START,END)
```

*Arguments*

| | | | |
|---|---|---|---|
| POINTS | INTEGER | IN | Number of points to be processed |
| P_LEVELS_MODEL | INTEGER | IN | Number of model levels |
| Q_LEVELS_MODEL | INTEGER | IN | Number of wet levels |
| L | INTEGER | IN | Reference level |
| P(POINTS) | REAL | IN | Pressure surface on which results required |
| PL(POINTS) | REAL | IN | Reference pressure at level L |
| PSTAR(POINTS) | REAL | IN | Surface pressure |
| P_EXNER_HALF(POINTS ,P_LEVELS+1) | REAL | IN | Exner pressure at model half levels |
| Z(POINTS) | REAL | OUT | Height of pressure surface P |
| ZH(POINTS,P_LEVELS) | REAL | IN | Height of model half levels |
| THETA(POINTS,P_LEVELS) | REAL | IN | Potential temperature at full levels |
| Q(POINTS,Q_LEVELS) | REAL | IN | Specific humidity at full levels |
| AKH(P_LEVELS+1) | REAL | IN | Hybrid coord A at half levels |
| BKH(P_LEVELS+1) | REAL | IN | Hybrid coord B at half levels |
| START | INTEGER | IN | First point to be processed in |

| | | | |
|---|---|---|---|
| | | | POINTS dimension |
| END | INTEGER | IN | Last point to be processed in POINTS dimension |

**SUBROUTINE V_INT_T**

*Description*

Calculates the temperature along an arbitrary pressure level using the technique described in section 3.3.1.

*Call*

```
      CALL V_INT_T(T,P,PL,PSTAR,P_EXNER_HALF,THETA,POINTS,P_LEVELS,L
     *,AKH,BKH,START,END)
```

*Arguments*

| | | | |
|---|---|---|---|
| POINTS | INTEGER | IN | Number of points to be processed |
| P_LEVELS | INTEGER | IN | Number of model levels |
| L | INTEGER | IN | Reference level |
| T(POINTS) | REAL | OUT | Temperature along input pressure surface |
| P(POINTS) | REAL | IN | Pressure surface on which results required |
| PL(POINTS) | REAL | IN | Reference pressure at level L |
| PSTAR(POINTS) | REAL | IN | Surface pressure |
| P_EXNER_HALF(POINTS ,P_LEVELS+1) | REAL | IN | Exner pressure at model half levels |
| THETA(POINTS,P_LEVELS) | REAL | IN | Potential temperature at full levels |
| AKH(P_LEVELS+1) | REAL | IN | Hybrid coord A at half levels |
| BKH(P_LEVELS+1) | REAL | IN | Hybrid coord B at half levels |
| START | INTEGER | IN | First point to be processed in POINTS dimension |
| END | INTEGER | IN | Last point to be processed in POINTS dimension |

**SUBROUTINE PMSL**

*Description*

Calculates mean sea level pressure using the formula given by equation (3.11).

*Call*

```
     CALL PMSL(P_MSL,PL,PSTAR,P_EXNER_HALF,THETA,Q,PHI_STAR,POINTS
    *,P_LEVELS,Q_LEVELS,L,AKH,BKH,START,END)
```

*Arguments*

| | | | |
|---|---|---|---|
| POINTS | INTEGER | IN | Number of points to be processed |
| P_LEVELS | INTEGER | IN | Number of model levels |
| Q_LEVELS | INTEGER | IN | Number of wet levels |
| L | INTEGER | IN | Reference level |
| P_MSL(POINTS) | REAL | OUT | Mean sea level pressure |
| PHI_STAR(POINTS) | REAL | IN | Geopotential height of topography |
| PL(POINTS) | REAL | IN | Reference pressure at level L |
| PSTAR(POINTS) | REAL | IN | Surface pressure |
| P_EXNER_HALF(POINTS ,P_LEVELS+1) | REAL | IN | Exner pressure at model half |

| | | | |
|---|---|---|---|
| | | | levels |
| THETA(POINTS,P_LEVELS) | REAL | IN | Potential temperature at full levels |
| Q(POINTS,P_LEVELS) | REAL | IN | Potential temperature at full levels |
| AKH(P_LEVELS+1) | REAL | IN | Hybrid coord A at half levels |
| BKH(P_LEVELS+1) | REAL | IN | Hybrid coord B at half levels |
| START | INTEGER | IN | First point to be processed in POINTS dimension |
| END | INTEGER | IN | Last point to be processed in POINTS dimension |

## SUBROUTINE TROP

*Description*

Calculates tropopause temperature, pressure and height using the technique described in section 3.3.2. At those points where the tropopause cannot be determined, a missing data indicator is returned.

*Call*

```
   CALL TROP(PSTAR,THETA,P_EXNER_HALF,ZH,TT,PT,ZT,POINTS,P_LEVELS
  &,MIN_TROP_LEVEL,AKH,BKH)
```

*Arguments*

| | | | |
|---|---|---|---|
| POINTS | INTEGER | IN | Number of points to be processed |
| P_LEVELS | INTEGER | IN | Number of model levels |
| MIN_TROP_LEVEL | INTEGER | IN | Level number for lowest possible trop. Set to first level above boundary layer. |
| PSTAR(POINTS) | REAL | IN | Surface pressure |
| THETA(POINTS,P_LEVELS) | REAL | IN | Potential temperature at full levels |
| P_EXNER_HALF(POINTS ,P_LEVELS+1) | REAL | IN | Exner pressure at model half levels |
| ZH(POINTS,P_LEVELS+1) | REAL | IN | Height of model half levels |
| TT(POINTS) | REAL | OUT | Temperature of tropopause |
| PT(POINTS) | REAL | OUT | Pressure of tropopause |
| ZT(POINTS) | REAL | OUT | Height of tropopause |
| AKH(P_LEVELS+1) | REAL | IN | Hybrid coord A at half levels |
| BKH(P_LEVELS+1) | REAL | IN | Hybrid coord B at half levels |

## SUBROUTINE VINT_TH

*Description*

Calculates potential temperature of model layers using temperatures input on an arbitrary set of pressure levels. The algorithm used is described in section 3.3.3.

*Call*

```
      CALL VINT_TH(P_HALF,P_EXNER_HALF,THETA,T_SRCE,POINTS,P_SRCE,P_LEVELS,
     * SRCE_LEVELS,PSTAR,AKH,BKH)
```

*Arguments*

| | | | |
|---|---|---|---|
| POINTS | INTEGER | IN | Number of points to be processed |
| P_LEVELS | INTEGER | IN | Number of model levels |
| SRCE_LEVELS | INTEGER | IN | Number of input levels |
| P_HALF(POINTS, P_LEVELS+1) | REAL | IN | Pressure of model half levels |
| P_EXNER_HALF(POINTS, P_LEVELS+1) | REAL | IN | Exner pressure at model half levels |
| THETA(POINTS,P_LEVELS) | REAL | OUT | Potential temperature at full levels |
| T_SRCE(POINTS, SRCE_LEVELS) | REAL | IN | Input temperature fields |
| P_SRCE(POINTS, SRCE_LEVELS) | REAL | IN | Pressure of input temperature fields |
| PSTAR(POINTS) | REAL | IN | Surface pressure |
| AKH(P_LEVELS+1) | REAL | IN | Hybrid coord A at half levels |
| BKH(P_LEVELS+1) | REAL | IN | Hybrid coord B at half levels |

## SUBROUTINE V_INT

*Description*

Performs vertical interpolation from one arbitrary set of pressure levels to another. The technique used is linear interpolation in log(p) (see section 3.1). When interpolating wind components, there is an option (controlled by MAX_WIND) for including data from max wind modelling.

*Call*

```
      CALL V_INT(P_IN,P_OUT,DATA_IN,DATA_OUT,POINTS,LEVELS
     *                ,DATA_MAXW,P_MAXW,MAX_WIND)
```

*Arguments*

| | | | |
|---|---|---|---|
| POINTS | INTEGER | IN | Number of points to be processed. |
| LEVELS | INTEGER | IN | Number of levels in source data. |
| P_IN(POINTS,LEVELS) | REAL | IN | 3-D pressure field of source data. |
| P_OUT(POINTS) | REAL | IN | Array of pressure values to be interpolated to. |
| DATA_IN(POINTS,LEVELS) | REAL | IN | Source data. |
| DATA_OUT(POINTS) | REAL | OUT | Result of interpolation. |
| DATA_MAXW(POINTS) | REAL | IN | Max wind data |
| P_MAXW(POINTS) | REAL | IN | Pressure of max wind data |
| MAX_WIND | LOGICAL | IN | Switch to include max winds if required. |

# TIME INTERPOLATION

## SUBROUTINE T_INT

*Description*

Carries out linear interpolation in time between two fields. If the missing data indicator is present at one of the times, the value at the other time is used.

*Call*

```
     CALL T_INT(DATA_T1,T1,DATA_T2,T2,DATA_T3,T3,POINTS)
```

*Arguments*

| | | | |
|---|---|---|---|
| POINTS | INTEGER | IN | Number of points to be processed |
| DATA_T1(POINTS) | REAL | IN | Data at time T1 |
| DATA_T2(POINTS) | REAL | IN | Data at time T2 |
| DATA_T3(POINTS) | REAL | OUT | Data at time T3 |
| T1 | REAL | IN | Time of first data field |
| T2 | REAL | IN | Time of second data field |
| T3 | REAL | IN | Time at which new field is required T1<=T3<=T2 |

## SUBROUTINE T_INT_C

*Description*

Carries out linear interpolation in time between two fields at times T1 and T2. If the missing data indicator is present at one of the times, the value at the other time is used. The interpolation is controlled by a field ZI. A prescribed value is inserted where ZI=0. If ZI changes between 0 and non-zero in the period T1 - T2, then the field is linearly interpolated between its value at the time when ZI is non-zero and the prescribed value at the time when ZI becomes zero. The fractional time at which ZI changes between 0 and non-zero in the period T1 - T2 must be provided as input. If ZI=MD, then linear interpolation is carried out using subroutine T_INT.

*Call*

```
      CALL T_INT_C(DATA_T1,T1,DATA_T2,T2,DATA_T3,T3,POINTS
     *,FRAC_TIME,ZI_T1,PRES_VALUE)
```

*Arguments*

| | | | |
|---|---|---|---|
| POINTS | INTEGER | IN | Number of points to be processed |
| DATA_T1(POINTS) | REAL | IN | Data at time T1 |
| DATA_T2(POINTS) | REAL | IN | Data at time T2 |
| DATA_T3(POINTS) | REAL | OUT | Data at time T3 |
| T1 | REAL | IN | Time of first data field |
| T2 | REAL | IN | Time of second data field |
| T3 | REAL | IN | Time at which new field is required T1<=T3<=T2 |
| ZI_T1(POINTS) | REAL | IN | Value of controlling field at T1 |
| PRES_VALUE(POINTS) | REAL | IN | Prescribed value of DATA when ZI=0 |
| FRAC_TIME(POINTS) | REAL | IN | Fractional time at which ZI changes between zero and non-zero in this time range |

## SUBROUTINE FRAC_TIM

*Description*

Calculates fractional time at which the control field ZI changes from zero to non-zero or visa versa. A missing data indicator is returned if no such change takes place. The algorithm assumes that the changes progress in the latitudinal direction from time T1 - T2. Used for snow depth and ice-fraction ancillary fields.

*Call*

```
      CALL FRAC_TIM(DATA_T1,DATA_T2,FRAC_TIME,P_ROWS,ROW_LENGTH)
```

| | | |
|---|---|---|
| ROW_LENGTH | INTEGER | IN Length of row |
| P_ROWS | INTEGER | IN Number of rows |
| DATA_T1(ROW_LENGTH,P_ROWS) | REAL | IN Data at time T1 |
| DATA_T2(ROW_LENGTH,P_ROWS) | REAL | IN Data at time T2, where T1<T2 |
| FRAC_TIME(ROW_LENGTH,P_ROWS) | REAL | OUT Fractional time at which DATA changes between zero and non-zero in this time range |

# RECONFIGURATION

*Librarian's note: all revision information must appear at the front of the entire paper S1.*

## 1.INTRODUCTION

The reconfiguration program has been written in a flexible way. It allows atmosphere model data to be interpolated to new resolutions and fields to be added to or subtracted from the dump. Ancillary data and separate upper air analyses may be incorporated into a dump provided that their files conform to the format described in UM Documentation Paper F3. An option to transplant sections of one dump into another is also provided. ECMWF analyses stored on pressure levels or ECMWF model levels may also be imported as GRIB code. A sample job to extract ECMWF analyses from the MARS archive is given in Section 4.

Ocean dumps may be processed so that fields may be added to or subtracted from the dump.

## 2.RUNNING THE PROGRAM

The functions of the program are controlled by NAMELIST inputs. These are described below. The User Interface provides access to these features in a user friendly manner.

The amount of memory required to run reconfiguration depends largely on the output resolution and the amount of ancillary data to be incorporated into the dump. On MPP machines, memory is distributed among the processors, only a small proportion being available to individual processors. This can cause problems in reconfiguring to high resolution as reconfiguration is not yet parallelized.

On the Met Office T3E, the reconfiguration executable is built using the *setlabel* command to force it to run on a high memory (64 MWord) processor instead of on a 16 MWord processor.

i.e. setlabel –l HHIGHMEM qxrecon_dump

Reconfiguration to climate resolution will currently fit on a 16 MWord processor, but reconfiguration to operational global forecast resolution requires a 64 MWord processor. Extra options, such as incorporating several multi-layer ancillary fields, may require more memory.

## NAMELIST RECON

*Purpose*

Sets up the dimensions of those arrays which store information at the output resolution. These values are also used to fill the appropriate entries in FIXHD_OUT and INTEGER_CONSTANTS_OUT. Note that the input dimensions are determined from the input header records.

*Variables*

| Name | Type | Description | Default |
|---|---|---|---|
| SCALE | I | Scale factor used to initialise aerosol increments in PFtoUM stage of VAR reconfiguration | 18 |
| SUBMODEL_IDENT | I | Submodel Identifier. | .FALSE. |
| DUMP_PACK | I | Packing Indicator | .FALSE. |
| LAND_POINTS_OUT | I | Number of land points on output grid | |
| ANVIL_FACTOR | R | Parameter required to calculate vertical cloud amount distribution | 1.0 |
| TOWER_FACTOR | R | Parameter required to calculate vertical cloud amount distribution | 1.0 |
| RIMWIDTHA | I | No of points width in atmosphere rim fields | |
| P_LEVELS_OUT | I | Number of Levels | P_LEVELS_IN |
| Q_LEVELS_OUT | I | Number of wet levels | Q_LEVELS_IN |
| P_ROWS_OUT | I | Number of P-rows | P_ROWS_IN |
| ROW_LENGTH_OUT | I | Row length | ROW_LENGTH_IN |
| TR_LEVELS_OUT | I | Number of tracer levels | TR_LEVELS_IN |
| TR_LEVELS_ADV_OUT | I | Number of tracer levels advected | TR_LEVELS_ADV_IN |
| TR_VARS_OUT | I | Number of tracer variables | TR_VARS_IN |
| ST_LEVELS_OUT | I | Number of soil temperature levels | DS_LEVELS_IN |
| SM_LEVELS_OUT | I | Number of soil moisture levels | SM_LEVELS_IN |
| BL_LEVELS_OUT | I | Number of boundary layer levels | BL_LEVELS_IN |
| OZONE_LEVELS_OUT | I | Number of ozone levels | OZONE_LEVELS_IN |
| LEN_FIXHD_OUT | I | Length of fixed length header | LEN_FIXHD_IN |
| LEN_INTHD_OUT | I | Length of integer constant block | LEN_INTH_IN |
| LEN_REALHD_OUT | I | Length of real constants block | LEN_REALHD_OUT |
| LEN2_LEVDEPC_OUT | I | 2nd dim of level dependent constants | LEN2_LEVDEPC_IN |
| LEN2_ROWDEPC_OUT | I | 2nd dim of row dependent consts | LEN2_ROWDEPC_IN |
| LEN2_COLDEPC_OUT | I | 2nd dim of column dependent constants | LEN2_COLDEPC_IN |
| LEN1_FLDDEPC_OUT | I | 1st dim of field dependent constants | LEN1_FLDDEPC_IN |
| LEN2_FLDDEPC_OUT | l | 2nd dim of field dependent constants | LEN2_FLDDEPC_IN |
| LEN_EXTCNST_OUT | I | Length of extra consts block | LEN_EXTCNST_IN |
| LEN_DUMPHIST_OUT | I | Length of history block | LEN_DUMPHIST_IN |
| LEN_CFI1_OUT | I | Length of compressed index 1 | LEN_CFI1_IN |

| | | | |
|---|---|---|---|
| LEN_CFI2_OUT | I | LENGTH of compressed index 2 | LEN_CFI2_IN |
| LEN_CFI3_OUT | I | Length of compressed index 3 | LEN_CFI3_IN |
| LEN1_LOOKUP_OUT | I | 1st dim of lookup table | LEN1_LOOKUP_IN |
| MAX_VARIABLES_OUT | I | Number of different field types | MAX_VARIABLES_IN |
| GRIB | L | ECMWF grib data used as input | .FALSE. |
| UARS | L | Upper air data to be imported | .FALSE. |
| RESET | L | Set data time to verification time in fixed length header: no interpolation is done if RESET=T | .FALSE. |
| PERTURBATION | R | Controls incorporation of ECMWF ensemble perturbations.<br>= 1.0 - add in perturbation<br>=-1.0 - subtract off perturbation | 0.0 |
| STRAT_Q | L | Reset stratospheric moisture to climatological values using subroutine STRATQ | .FALSE. |
| TRANS | L | Transplant switch | .FALSE. |
| SPIRAL_S | L | Use spiral coastal adjustment | .FALSE. |
| LCAL360 | L | Use 360 day calendar | .FALSE. |
| LOZONE_ZONAL | L | Use zonal ozone fields | .FALSE. |
| LAMIPII | L | True if AMIP II run | .FALSE. |

## NAMELIST NSUBMODL

*Purpose*

Specify submodel and relevant internal models

*Variables*

| Name | Type | Description |
|---|---|---|
| N_INTERNAL_MODEL | I | Number of internal models |
| INTERNAL_MODEL_LIST | I | List of internal models |
| SUBMODEL_FOR_IM | I | Submodel |

## NAMELIST USTSNUM

*Purpose*

Specify userSTASHmaster files

*Variables*

| Name | Type | Description |
|---|---|---|
| N_USTASH | I | Number of userSTASHmasters |
| NRECS_USTASH | I | Total number of userSTASHmaster records |
| USTSFILS | C | Location of userSTASHmasters |

## NAMELIST VERTICAL

*Purpose*

Controls vertical interpolation by inputting Eta values at model half levels

*Variables*

| Name | Type | Description | Default |
|---|---|---|---|
| METH_LEV_CALC | I | Method of calculating Eta values at model levels | 5 |
| ETAH(1000) | R | Eta values at model layer boundaries | RMDI |
| MIN_PRS_HLEV | I | Level above which pressure coordinates used | 0 |
| MAX_SIG_HLEV | I | Level below which sigma coordinates used | 0 |

## NAMELIST HORIZONT

*Purpose*

Controls horizontal interpolation and transformation to limited area rotated grid.

*Variables*

| Name | Type | Description | Default |
|---|---|---|---|
| GLOBAL | L | .T. for global model | .TRUE. |
| H_INT_TYPE | L | .T. for area weighted interpolation<br>.F. for bi-linear interpolation | .FALSE. |
| LPOLARCHK | L | .T. if non-constant polar rows are to be corrected for fields on p-grid | .TRUE. |
| DELTA_LAMBDA | R | E_W grid length RELHD_OUT(1) | RMDI |
| DELTA_PHI | R | N-S grid length RELHD_OUT(2) | RMDI |
| LAMBDA_TLC | R | Longitude of 1st pt in row RELHD_OUT(4) | 0. |
| PHI_TLC | R | Latitude of 1st row RELHD_OUT(3) | 90. |
| LAMBDA_NPOLE | R | Longitude of north pole RELHD_OUT(6) | 0. |
| PHI_NPOLE | R | Latitude of north pole RELHD_OUT(5) | 90. |
| IPROJ | I | Projection number FIXHD_OUT(11) | IMDI |
| OCEAN_BOUNDARY_CONDITIONS | I | Ocean bound conds indicator FIXHD_OUT(11) | 0 |
| OCEAN_DYNAMICS | I | Ocean dynamics indicator FIXHD_OUT(12) | 0 |
| OCEAN_SEA_POINTS | I | Number of ocean sea points INTHD_OUT(11) | 0 |
| RIM_WIDTH_NORTH_TOPOG )<br>RIM_WIDTH_SOUTH_TOPOG }<br>RIM_WIDTH_EAST_TOPOG )<br>RIM_WIDTH_WEST_TOPOG ) | I | Width of the border specifying the rectangular sub-area in which the orography is set from ancillary data. Used in limited area versions.<br>Default 0 , ie full area | . |

## NAMELIST HEADERS

*Purpose*

Overrides values set by reconfiguration in fixed length, integer and real headers.

*Variables*

| Name | Type | Description | Default |
|---|---|---|---|
| FIXHD(256) | I | Fixed length header | No change |
| INTHD(100) | I | Integer constants header | No change |
| RELHD(100) | R | Real constants header | No change |

## NAMELIST ITEMS

*Purpose*

Controls the initialisation of fields not to be copied from the input dump.

*Variables*

| Name | Type | Description | Default |
|---|---|---|---|
| ITEM | I | Item code of field | 0 |
| SOURCE | I | Source of data<br>1= input dump<br>2= ancillary field<br>3= set data to 0/0.0/F<br>4= set to RMDI/IMDI<br>5= input from tracer file<br>6= set to constant<br>7= set to field from external dump<br>8= Initialize from other fields in input dump. Currently only slab temperature is initialised this way though the namelist.<br>9= Used in VAR reconfiguration to initialize u, v, $\theta_l$ and qt on the Charney-Phillips grid from the equivalent field on the UM grid and vice-versa | 0 |
| DOMAIN | I | Area covered 1= full output area<br>2= sub area | 0 |
| USER_PROG_RCONST | R | Constant value for user prognostic | 0.0 |
| USER_PROG_ANCIL_FILE | C | External file name containing the field to be used to initialize user prognostic | ' ' |
| USER_PROG_ANCIL_ITEMC | I | Item code of field in above file to be used to initialize user prognostic | ITEM |

## TRANS

*Purpose*

Specifies fields and sub areas to be transposed from one dump into another.

*Variables*

| Name | Type | Description | Default |
|---|---|---|---|
| ITEMC | I | Item code | 0 |
| LEVEL1 | I | Lowest level in range | 1 |
| LEVEL2 | I | Highest level in range | P_LEVELS_OUT |
| ROW1 | I | First row in range | 1 |
| ROW2 | I | Last row in range | P_ROWS_OUT |
| COL1 | I | First column in range | 1 |
| COL2 | I | Last column in range | ROW_LENGTH_OUT |

Other namelists are used by the addressing, but cover a far wider area than reconfiguration and are not covered here.

EXAMPLES

```
# Global -> Mesoscale with vertical interpolation and ancillary fields.
 &RECON
 SCALE=18, SUBMODEL_IDENT=1, DUMP_PACK=1, LAND_POINTS_OUT=10811,
 TOWER_FACTOR=0.0, ANVIL_FACTOR=0.0, RIMWIDTHA=4,
 P_LEVELS_OUT= 38, Q_LEVELS_OUT= 35, P_ROWS_OUT= 182, ROW_LENGTH_OUT= 146,
 TR_LEVELS_OUT= 0, TR_VARS_OUT= 0, ST_LEVELS_OUT=3, SM_LEVELS_OUT=0,
 BL_LEVELS_OUT= 14, OZONE_LEVELS_OUT=11,
 LEN_FIXHD_OUT=256, LEN_INTHD_OUT=29, LEN_REALHD_OUT=38,
 LEN2_LEVDEPC_OUT=6, LEN2_ROWDEPC_OUT=3, LEN2_COLDEPC_OUT=0,
 LEN2_FLDDEPC_OUT=0, LEN_EXTCNST_OUT=0, LEN_DUMPHIST_OUT=0,
 MAX_VARIABLES_OUT= -32768,
 GRIB= .FALSE., UARS= .FALSE., RESET= .FALSE., PERTURBATION= 0.0,
 TRANS= .FALSE., LSPIRAL_S= .TRUE., LCAL360=.FALSE.,  LOZONE_ZONAL=.FALSE.,
 LAMIPII= .FALSE.
 &END

 &NSUBMODL
 N_INTERNAL_MODEL=1,
 INTERNAL_MODEL_LIST= 1 ,
 SUBMODEL_FOR_IM= 1 ,
 &END

 &USTSNUM
 N_USTASH= 1 , NRECS_USTASH= 0 ,
 USTSFILS= "PRESM_A" &END

 &VERTICAL
 METH_LEV_CALC= 5,
 ETAH= 1.000000000000,0.997600000000,0.992900000000,0.983500000000,
       0.971900000000,0.958000000000,0.940000000000,0.921000000000,
       0.901000000000,0.880000000000,0.858000000000,0.835000000000,
       0.810000000000,0.780000000000,0.745000000000,0.705000000000,
       0.660000000000,0.610000000000,0.555000000000,0.500000000000,
       0.450000000000,0.410000000000,0.370000000000,0.340000000000,
```

```
       0.315000000000,0.290000000000,0.265000000000,0.240000000000,
       0.215000000000,0.190000000000,0.165000000000,0.140000000000,
       0.115000000000,0.090000000000,0.065000000000,0.040000000000,
       0.020000000000,0.010000000000, 0.000500000000,
  MIN_PRS_HLEV= 36, MAX_SIG_HLEV= 12,
 &END

 &HORIZONT
 GLOBAL= .FALSE.,
 DELTA_LAMBDA= 0.110000,  DELTA_PHI= 0.110000,
 LAMBDA_TLC=   352.9500, PHI_TLC=    11.5000,
 LAMBDA_NPOLE=   177.5000, PHI_NPOLE=    37.5000,
 RIM_WIDTH_NORTH_TOPOG=8,
 RIM_WIDTH_SOUTH_TOPOG=8,
 RIM_WIDTH_WEST_TOPOG=8,
 RIM_WIDTH_EAST_TOPOG=8,
 H_INT_TYPE= .FALSE.,
 &END

 &HEADERS
 FIXHD(12)=405,
 &END

 # &STSHCOMP
 RUN_TARGET_END= 0 , 0 , 0 , 3 , 0 , 0 ,
 OCAAA=2,
 INDEP_SR=’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,
 ’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’0A’,’  ’,’  ’,’  ’,’
’,
 ’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’
’,
 ’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’
’,
 ’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’
’,
 ’  ’,’1A’,’  ’,’0A’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’1A’,’  ’,’1A’,’
’,
 ’1A’,’  ’,’  ’,’  ’,’  ’,’  ’,’2B’,’1C’,’1A’,’1A’,’1A’,’2A’,’0A’,’1A’,’
’,
 ’0A’,
 EWSPACEA=0.110000, NSSPACEA=0.110000,
 FRSTLATA=   11.5000, FRSTLONA=  352.9500,
 POLELATA=   37.5000, POLELONA=  177.5000,
 MESO=’Y’,
 SWBND=4, LWBND=6,
 SWINC=24, LWINC=24,
 OROGR=’Y’,
 SWMCR=’’,
 ATMOS_SR=’  ’,’1B’,’1A’,’3A’,’2E’,’3C’,’0A’,’0A’,’1A’,
 ’1A’,’1C’,’1A’,’1B’,’1C’,’0A’,’1A’,’1A’,’0A’,’0A’,’0A’,’
’,’0A’,’0A’,’0A’,
 ’0A’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’
’,
 ’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’
’,
 ’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’
’,
 ’  ’,’0A’,’1A’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’
```

```
’,
 ’  ’,’  ’,’  ’,’1A’,
 ZonAvOzone=.FALSE.,
 StLevGWdrag=, BotVDiffLev=, TopVDiffLev=,
 OCALB=1,
 FLOOR=’N’,
 TOTAE=’Y’, TOTEM=’Y’,
 TCA=0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
 0,0,0,0,0,
 SSTAnom=’N’,
 &END

&NLSTCATM
 L_SSICE_ALBEDO=.FALSE.,
 L_VINT_TP=.FALSE.,
 L_RHCPT=.FALSE.,
 L_CLD_AREA=.FALSE.,
 L_CO2_INTERACTIVE=.FALSE.,
 L_CO2_EMITS= .FALSE.,
 L_VEG_FRACS=.FALSE.,
 L_TRIFFID=.FALSE.,
 L_PHENOL=.FALSE.,
 L_TRIF_EQ=.FALSE.,
 L_NRUN_MID_TRIF=.FALSE.,
 L_3D_CCA=.FALSE.,
 L_CLOUD_DEEP=.FALSE.,
 L_CCW=.FALSE.,
 L_PHASE_LIM=.FALSE.,
 L_LSPICE=.FALSE.,
 L_LSPICE_BDY=.FALSE.,
 L_BL_LSPICE=.FALSE.,
 L_SNOW_ALBEDO=.FALSE.,
 H_SWBANDS= 4,
 H_LWBANDS= 6,
 A_SWEEPS_DYN= 4,
 A_ADJSTEPS= 3,
 A_SW_RADSTEP=12,
 A_SW_SEGMENTS=1,
 A_LW_RADSTEP=12,
 A_LW_SEGMENTS=1,
 A_CONV_STEP=1,
 A_CONVECT_SEGMENTS=4,
 L_RMBL=.TRUE.,
 L_MIXLEN=.FALSE.,
 A_ENERGYSTEPS=0,
 A_NSET_FILTER=72,
 A_ASSIM_MODE=’NONE’,
 LEXPAND_OZONE=.FALSE.,
 L_NEG_THETA=.FALSE.,
 L_NEG_PSTAR=.FALSE.,
 L_NEG_QT=.TRUE.,
 L_NEG_TSTAR=.FALSE.,
 L_FIELD_FLT=.FALSE.,
 L_Z0_OROG=.TRUE.,
 L_HALF_TIMESTEP_DYN=.TRUE.,
 L_SUPERBEE=.TRUE.,
 L_TRACER_THETAL_QT=.FALSE.,
 L_HALF_TIMESTEP_DIV=.TRUE.,
```

```
 L_QT_POS_LOCAL=.TRUE.,
 LLINTS=.TRUE.,
 LWHITBROM=.TRUE.,
 LGWLINP=.FALSE.,
 LEMCORR=.FALSE.,
 L_MURK=.TRUE.,
 L_BL_TRACER_MIX=.FALSE.,
 L_MOM=.FALSE.,
 L_CAPE=.FALSE.,
 L_SDXS=.TRUE.,
 L_XSCOMP=.TRUE.,
 L_MURK_ADVECT=.TRUE.,
 L_MURK_SOURCE=.TRUE.,
 L_MURK_BDRY=.TRUE.,
 LMICROPHY=.FALSE.,
 L_SULPC_SO2=.FALSE.
 L_SOOT=.FALSE.
 L_SO2_SURFEM=.FALSE.,
 L_SO2_HILEM=.FALSE.,
 L_SO2_NATEM=.FALSE.,
 L_SULPC_DMS=.FALSE.,
 L_DMS_EM=.FALSE.,
 L_USE_SULPC_DIRECT=.FALSE.,
 L_SULPC_OZONE=.FALSE.,
 L_SULPC_NH3=.FALSE.,
 L_NH3_EM=.FALSE.,
 L_USE_SULPC_INDIRECT_SW=.FALSE.,
 L_USE_SULPC_INDIRECT_LW=.FALSE.,
 L_SOOT_SUREM=.FALSE.,
 L_SOOT_HILEM=.FALSE.,
 L_USE_SOOT_DIRECT=.FALSE.,
 L_CLIMAT_AEROSOL=.FALSE.,
 LSINGLE_HYDROL=.TRUE.,
 LMOSES=.FALSE.,
 L_H2_SULPH=.FALSE.,
 &END
&NLSTCSLB &END

 ### Ancillary fields ###
 &ITEMS ITEM=30, DOMAIN=1, SOURCE=2, &END
 &ITEMS ITEM=33, DOMAIN=1, SOURCE=2, &END
 &ITEMS ITEM=34, DOMAIN=1, SOURCE=2, &END
 &ITEMS ITEM=35, DOMAIN=1, SOURCE=2, &END
 &ITEMS ITEM=36, DOMAIN=1, SOURCE=2, &END
 &ITEMS ITEM=37, DOMAIN=1, SOURCE=2, &END
 &ITEMS ITEM=60, DOMAIN=1, SOURCE=2, &END
 &ITEMS ITEM=21, DOMAIN=1, SOURCE=2, &END
 &ITEMS ITEM=23, DOMAIN=1, SOURCE=2, &END
 &ITEMS ITEM=20, DOMAIN=1, SOURCE=2, &END
 &ITEMS ITEM=40, DOMAIN=1, SOURCE=2, &END
 &ITEMS ITEM=41, DOMAIN=1, SOURCE=2, &END
 &ITEMS ITEM=42, DOMAIN=1, SOURCE=2, &END
 &ITEMS ITEM=43, DOMAIN=1, SOURCE=2, &END
 &ITEMS ITEM=44, DOMAIN=1, SOURCE=2, &END
 &ITEMS ITEM=45, DOMAIN=1, SOURCE=2, &END
 &ITEMS ITEM=46, DOMAIN=1, SOURCE=2, &END
 &ITEMS ITEM=47, DOMAIN=1, SOURCE=2, &END
 &ITEMS ITEM=50, DOMAIN=1, SOURCE=2, &END
```

```
 &ITEMS ITEM=51, DOMAIN=1, SOURCE=2, &END
 &ITEMS ITEM=52, DOMAIN=1, SOURCE=2, &END
 &ITEMS ITEM=53, DOMAIN=1, SOURCE=2, &END
 &ITEMS ITEM=54, DOMAIN=1, SOURCE=2, &END
 &ITEMS ITEM=55, DOMAIN=1, SOURCE=2, &END
 &ITEMS ITEM=56, DOMAIN=1, SOURCE=2, &END
 &ITEMS ITEM=26, DOMAIN=1, SOURCE=2, &END
 &ITEMS ITEM=31, DOMAIN=1, SOURCE=2, &END
 &ITEMS ITEM=24, DOMAIN=1, SOURCE=2, &END
 &ITEMS ITEM=32, DOMAIN=1, SOURCE=2, &END
 &ITEMS ITEM=19, DOMAIN=1, SOURCE=2, &END
 &ITEMS ITEM=57, DOMAIN=1, SOURCE=2, &END
 &ITEMS ITEM=90, DOMAIN=1, SOURCE=2, &END
 &ITEMS ITEM=17, DOMAIN=1, SOURCE=2, &END
 &ITEMS ITEM=18, DOMAIN=1, SOURCE=2, &END
 ### Lateral Bounday tendancies fields ###
 &ITEMS ITEM=96, DOMAIN=2, SOURCE=4, &END
 &ITEMS ITEM=97, DOMAIN=2, SOURCE=4, &END
 ### Transplant data ####


## END OF FILE ###

#Global to climate adding user prognostics
 &RECON
 SCALE=18,SUBMODEL_IDENT=1, DUMP_PACK=1, LAND_POINTS_OUT=2381,
 ANVIL_FACTOR=0.0, TOWER_FACTOR=0.0, RIMWIDTHA=1,
 P_LEVELS_OUT= 19, Q_LEVELS_OUT= 16, P_ROWS_OUT= 73, ROW_LENGTH_OUT= 96,
 TR_LEVELS_OUT= 0, TR_VARS_OUT= 0, ST_LEVELS_OUT=3, SM_LEVELS_OUT=0,
 BL_LEVELS_OUT= 5, OZONE_LEVELS_OUT=9,
 LEN_FIXHD_OUT=256, LEN_INTHD_OUT=29, LEN_REALHD_OUT=38
 LEN2_LEVDEPC_OUT=6, LEN2_ROWDEPC_OUT=3, LEN2_COLDEPC_OUT=0,
 LEN2_FLDDEPC_OUT=0, LEN_EXTCNST_OUT=0, LEN_DUMPHIST_OUT=0,
 MAX_VARIABLES_OUT= -32768,
 GRIB= .FALSE., UARS= .FALSE., RESET= .FALSE., PERTURBATION= 0.0,
 TRANS= .FALSE., LSPIRAL_S= .FALSE., LCAL360=.FALSE., LOZONE_ZONAL=.TRUE.,
 LAMIPII= .FALSE.
 &END

 &NSUBMODL
 N_INTERNAL_MODEL=1,
 INTERNAL_MODEL_LIST= 1 ,
 SUBMODEL_FOR_IM= 1 ,
 &END

 &USTSNUM
 N_USTASH= 1 , NRECS_USTASH= 4 ,
 USTSFILS= "PRESM_A" &END

 &VERTICAL
 METH_LEV_CALC= 5,
 ETAH= 1.000000000000,0.994000000000,0.956000000000,0.905000000000,
       0.835000000000,0.750000000000,0.650000000000,0.550000000000,
       0.460000000000,0.385000000000,0.325000000000,0.275000000000,
       0.225000000000,0.175000000000,0.125000000000,0.075000000000,
       0.040000000000,0.020000000000,0.010000000000,0.000500000000,
 MIN_PRS_HLEV= 17, MAX_SIG_HLEV= 5,
 &END
```

```
 &HORIZONT
 GLOBAL= .TRUE.,
 RIM_WIDTH_NORTH_TOPOG=0,
 RIM_WIDTH_SOUTH_TOPOG=0,
 RIM_WIDTH_WEST_TOPOG=0,
 RIM_WIDTH_EAST_TOPOG=0,
 H_INT_TYPE= .FALSE.,
 &END

 &HEADERS
 FIXHD(12)=405,
 &END

 &STSHCOMP
 RUN_TARGET_END= 0 , 0 , 0 , 6 , 0 , 0 ,
 OCAAA=1,
 INDEP_SR=’1A’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,
 ’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’1A’,’  ’,’  ’,’  ’,
 ’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,
 ’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,
 ’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,
 ’  ’,’  ’,’  ’,’  ’,’  ’,’1A’,’  ’,’0A’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,
 ’  ’,’1A’,’  ’,’1A’,’  ’,’1A’,’  ’,’  ’,’  ’,’  ’,’  ’,’2B’,’1C’,’1A’,
 ’1A’,’1A’,’2A’,’0A’,’1A’,’  ’,’0A’,
 SWBND=4, LWBND=6,
 SWINC=8, LWINC=8,
 OROGR=’Y’,
 SWMCR=’N’,
 ATMOS_SR=’  ’,’2A’,’1B’,’3A’,’2C’,’3C’,’3A’,’1A’,’1A’,
 ’1A’,’1A’,’0A’,’1B’,’1A’,’0A’,’1A’,’1A’,’0A’,’1A’,’0A’,’
’,’1A’,’1A’,’1A’,
 ’1A’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’
’,
 ’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’
’,
 ’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’
’,
 ’  ’,’0A’,’1A’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’  ’,’
’,
 ’  ’,’  ’,’  ’,’1A’,
 ZonAvOzone=.TRUE.,
 StLevGWdrag=3, BotVDiffLev=8, TopVDiffLev=14,
 OCALB=1,
 FLOOR=’N’,
 TOTAE=’N’, TOTEM=’N’,
 TCA=0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
 0,0,0,0,0,
 SSTAnom=’N’,
 &END
 &NLSTCATM
 L_SSICE_ALBEDO=.FALSE.,
 L_VINT_TP=.FALSE.,
 L_RHCPT=.FALSE.,
 L_CLD_AREA=.FALSE.,
 L_CO2_INTERACTIVE=.FALSE.,
 L_CO2_EMITS= .FALSE.,
 L_VEG_FRACS=.FALSE.,
 L_TRIFFID=.FALSE.,
```

```
L_PHENOL=.FALSE.,
L_TRIF_EQ=.FALSE.,
L_NRUN_MID_TRIF=.FALSE.,
L_3D_CCA=.FALSE.,
L_CLOUD_DEEP=.FALSE.,
L_CCW=.FALSE.,
L_PHASE_LIM=.FALSE.,
L_LSPICE=.FALSE.,
L_LSPICE_BDY=.FALSE.,
L_BL_LSPICE=.FALSE.,
L_SNOW_ALBEDO=.FALSE.,
H_SWBANDS= 4,
H_LWBANDS= 6,
A_SWEEPS_DYN= 1,
A_ADJSTEPS= 3,
A_SW_RADSTEP=9,
A_SW_SEGMENTS=4,
A_LW_RADSTEP=9,
A_LW_SEGMENTS=8,
A_CONV_STEP=1,
A_CONVECT_SEGMENTS=2,
L_RMBL=.TRUE.,
L_MIXLEN=.TRUE.,
A_ENERGYSTEPS=0,
A_NSET_FILTER=18,
A_ASSIM_MODE='NONE',
LEXPAND_OZONE=.TRUE.,
L_NEG_THETA=.TRUE.,
L_NEG_PSTAR=.TRUE.,
L_NEG_QT=.TRUE.,
L_NEG_TSTAR=.FALSE.,
L_FIELD_FLT=.FALSE.,
L_Z0_OROG=.TRUE.,
L_HALF_TIMESTEP_DYN=.FALSE.,
L_TRACER_THETAL_QT=.FALSE. ,
L_HALF_TIMESTEP_DIV=.FALSE.,
L_QT_POS_LOCAL=.FALSE.,
LLINTS=.TRUE.,
LWHITBROM=.TRUE.,
LGWLINP=.FALSE.,
LEMCORR=.FALSE.,
L_MURK=.FALSE.,
L_BL_TRACER_MIX=.FALSE
L_MOM=.TRUE.
L_CAPE=.FALSE.,
L_SDXS=.FALSE.,
L_XSCOMP=.FALSE.,
L_MURK_ADVECT=.FALSE.,
L_MURK_SOURCE=.FALSE.,
L_MURK_BDRY=.FALSE.,
LMICROPHY=.FALSE.,
L_SULPC_SO2=.FALSE.
L_SOOT=.FALSE.
L_SO2_SURFEM=.FALSE.,
L_SO2_HILEM=.FALSE.,
L_SO2_NATEM=.FALSE.,
L_SULPC_DMS=.FALSE.,
L_DMS_EM=.FALSE.,
```

```
 L_USE_SULPC_DIRECT=.FALSE.,
 L_SULPC_OZONE=.FALSE.,
 L_SULPC_NH3=.FALSE.,
 L_NH3_EM=.FALSE.,
 L_USE_SULPC_INDIRECT_SW=.FALSE.,
 L_USE_SULPC_INDIRECT_LW=.FALSE.,
 L_SOOT_SUREM=.FALSE.,
 L_SOOT_HILEM=.FALSE.,
 L_USE_SOOT_DIRECT=.FALSE.,
 L_CLIMAT_AEROSOL=.FALSE.,
 LSINGLE_HYDROL=.TRUE.,
 LMOSES=.FALSE.,
 L_H2_SULPH=.FALSE.,
 &END
 &NLSTCSLB &END

 ### Ancillary fields ###
 &ITEMS ITEM=30, DOMAIN=1, SOURCE=2, &END
 &ITEMS ITEM=33, DOMAIN=1, SOURCE=2, &END
 &ITEMS ITEM=34, DOMAIN=1, SOURCE=2, &END
 &ITEMS ITEM=35, DOMAIN=1, SOURCE=2, &END
 &ITEMS ITEM=36, DOMAIN=1, SOURCE=2, &END
 &ITEMS ITEM=37, DOMAIN=1, SOURCE=2, &END
 &ITEMS ITEM=40, DOMAIN=1, SOURCE=2, &END
 &ITEMS ITEM=41, DOMAIN=1, SOURCE=2, &END
 &ITEMS ITEM=42, DOMAIN=1, SOURCE=2, &END
 &ITEMS ITEM=43, DOMAIN=1, SOURCE=2, &END
 &ITEMS ITEM=44, DOMAIN=1, SOURCE=2, &END
 &ITEMS ITEM=45, DOMAIN=1, SOURCE=2, &END
 &ITEMS ITEM=46, DOMAIN=1, SOURCE=2, &END
 &ITEMS ITEM=47, DOMAIN=1, SOURCE=2, &END
 &ITEMS ITEM=50, DOMAIN=1, SOURCE=2, &END
 &ITEMS ITEM=51, DOMAIN=1, SOURCE=2, &END
 &ITEMS ITEM=52, DOMAIN=1, SOURCE=2, &END
 &ITEMS ITEM=53, DOMAIN=1, SOURCE=2, &END
 &ITEMS ITEM=54, DOMAIN=1, SOURCE=2, &END
 &ITEMS ITEM=55, DOMAIN=1, SOURCE=2, &END
 &ITEMS ITEM=56, DOMAIN=1, SOURCE=2, &END
 &ITEMS ITEM=26, DOMAIN=1, SOURCE=2, &END
 &ITEMS ITEM=31, DOMAIN=1, SOURCE=2, &END
 &ITEMS ITEM=24, DOMAIN=1, SOURCE=2, &END
 &ITEMS ITEM=32, DOMAIN=1, SOURCE=2, &END
 &ITEMS ITEM=19, DOMAIN=1, SOURCE=2, &END
 &ITEMS ITEM=17, DOMAIN=1, SOURCE=2, &END
 &ITEMS ITEM=18, DOMAIN=1, SOURCE=2, &END
 ### Atmos user-prognostic fields ###
 &ITEMS ITEM=241, DOMAIN=1, SOURCE=6, USER_PROG_RCONST=4.400000e+00 &END
 &ITEMS ITEM=242, DOMAIN=1, SOURCE=7, USER_PROG_ANCIL_ITEMC=5
 USER_PROG_ANCIL_FILE="/u/m11/user3/t11dg/dumps_T3E/vn4.4/atmos/userprog"
&END
 &ITEMS ITEM=243, DOMAIN=1, SOURCE=6, USER_PROG_RCONST=4.000000e+00 &END
 ### Transplant data ####


 ### END OF FILE ###
```

## 3. DATA FILES

The following unit numbers are used by the reconfiguration program. All files are unstructured.

| UNIT NUMBER | CONTENTS |
|---|---|
| 2 | User Stashmaster file |
| 18 | ECMWF grib code |
| 19 | Intermediate file used when processing grib code data or when unit 20 does not contain $\theta_L$ or $q_T$ |
| 20 | Atmosphere source dump |
| 21 | Atmosphere output dump |
| 22 | Stashmaster file |
| 29 | Upper air analyses (UARS) |
| 30 | Ozone |
| 31 | Soil moisture and snow depth |
| 32 | Deep soil temperatures |
| 34 | Vegetation types |
| 35 | Sea surface temperatures |
| 36 | Sea-ice fields |
| 37 | ECMWF perturbations |
| 38 | Sea surface currents |
| 39 | Land sea mask |
| 40 | Ocean source dump |
| 41 | Ocean output dump |
| 48 | Atmosphere tracers |
| 49 | Ocean tracers |
| 59 | Heat Convergence (SLAB) |
| 96 | Orographic fields |
| 97 | Transplant fields |
| 109 | Murkiness |
| 110 | Single level sulphur emissions |
| 111 | Single level user ancillary fields |
| 112 | Multi-level user ancillary fields |
| 115 | Natural S02 emissions |
| 116 | Chemistry oxidants |
| 117 | Sulphate aerosol forcing |
| 118 | Surface C02 emissions |
| 135 | Initial fractions of surface types |
| 136 | Initial vegetation state |
| 137 | Disturbed fraction of vegetation |
| 139 | Soot emissions |

## 4. OBTAINING ECMWF ANALYSES

ECMWF analyses may be obtained by running the following jobs for a programmer with a workstation or T3E userid of user_id and a using an ECMWF userid xxx and password yyyyyy. Only those programmers registered under an ECMWF project can access this data. ip_address is the ip address of the destination machine. For NWP, it is 151.170.5.6 (fr0110). *marsfile* contains the data to be transferred and *gribfile* is the file name on the destination machine to which it is to be copied.

```
#QSUB/USER="xxx"/PASSWORD="yyyyyy"
#QSUB/CPUTIME_REQ=60
#QSUB
set -xS
cd TMPDIR
#
# Access MARS data under UNICOS
#
mars <<EOR
retrieve,type=an,levtype=sfc,levelist=off,
 repres=gg,date=901226,time=00,target="marsfile",
 param=st/sp/lsm/z,format=packed,grid=1.875/1.25,accuracy=normal.
retrieve,type=an,levtype=pl,
 levelist=all,
 repres=sh,date=901226,time=00,target="marsfile",param=t/u/v/r,
 format=packed,grid=1.875/1.25,accuracy=normal.
end
EOR
#
# Send MARS data down link to COSMOS
#
eccopy -u user_id -h ip_address -f gribfile marsfile
```

The job accesses the pressure level archive at ECMWF via the Met Office-ECMWF link and transfers the data back to a workstation or T3E at the Met Office (The horizontal grid resolution and the analysis time and date may be changed via the parameters *grid, time* and *date* respectively. In addition, for dates prior to 15/7/86 where the surface geopotential is unavailable, *param=st/msl/lsm* must be coded for the surface field extraction.

If data on ECMWF hybrid vertical coordinates is required, then *levtype=ml* and *param=t/u/v/q* should be coded for upper air fields. Note that this option will not work if surface pressure and surface geopotential are not available.

There is also a method for transferring analyses to COSMOS which is used operationally, but this is not described here.

a

## APPENDIX 3 to S1 ( from FR DIVISION WORKING PAPER NO 150 )

# A NEW ALGORITHM FOR POTENTIAL TEMPERATURE TO TEMPERATURE CONVERSION FOR STANDARD LEVEL OUTPUT

C Wilson

7 April 1993

## Introduction

This is a brief note proposing a new algorithm for the output of temperature at standard pressure levels from the unified model. The new algorithm is also appropriate for the derivation of background temperature profiles used in the assimilation of locally retrieved satellite data (LASS/GLOSS).

Swinbank and Wilson ( S Division Technical Note No. 48, 1990) investigated the vertical interpolation of temperature observations and model data and proposed consistent procedures based on the original choice of location of the nominal model levels as the central value of Exner pressure within a layer. This specification of Exner value at the same level as the model potential temperature enables conversion to temperature and a simple local vertical interpolation to be used for derivation of temperatures at standard pressure levels. The location of the nominal levels and specification of Exner pressure at layer "centres" has since been modified to be more consistent with the model geopotential equation and dynamical formulation. The relocated levels are lower (higher pressure) and give generally warmer temperatures, especially in the upper troposphere and stratosphere. The original formulation also resulted in a positive bias in the stratosphere but of a smaller magnitude. The bias for both formulations increases with height and is proportional to the thickness of model layers. With the reduction from 20 to 19 model layers and consequent increase in top layer thickness the error was made worse. For these reasons the question of temperature output modelling was reconsidered and the new algorithm described below was derived. The approach is to keep the internal model formulation unchanged but use a different Exner pressure formulation at model layer "centres" to convert from the model layer mean potential temperatures to temperatures at the nominal levels; interpolation from these temperatures to required output pressure levels is then linear in height as described in Swinbank and Wilson and Unified Documentation Paper S1. Alternative interpolation techniques such as cubic splines could be considered but we have chosen here to retain a local interpolation.

## 2) The problem

The procedure for obtaining temperatures (or heights) at an arbitrary pressure level from a numerical model which has only a finite number of layers should be as consistent as possible with both the model formulation and the observed data used in the model analysis. It is worth recalling some of the points made by Swinbank and Wilson since many of the possible procedures can lead to pitfalls:

> In numerical weather prediction models it is not always immediately obvious how the temperature (or potential temperature) is defined, and how they should be related to observed temperature profiles.
>
> The average temperature between two pressure levels is not a well-defined quantity; however in numerical models one uses a single temperature for a model layer (or level).
>
> It is also important to ensure that the vertical interpolation used for calculating observation-analysis statistics is consistent with that used in the analysis scheme. It is possible that apparent biases may be the result of the method of calculating the statistics rather

than a real bias between observations and model data.

The fundamental difficulty in the unified model is the interpretation of a model layer "average" potential temperature and assignment of a nominal level. The sensitivity to the exact location of the nominal level is due to the use of potential temperature as a model variable, particularly in the stratosphere. Using temperature as the model variable would help the output problem but would lead to similar difficulties within the model for considerations of static stability etc.

### 3) Model level assignment
### a) Current procedure

The model divides the atmosphere into a set of layers; the layer boundary locations are fundamental and are specified. Observed temperature profiles (as measured by sondes) are vertically interpolated to provide layer mean potential temperatures by matching the geopotential thickness from the observations to the form used by the model i.e.
Given a set of observed temperatures $\{ T_i \}$ between a model layer with boundaries at $p_{k+1/2}$ and $p_{k-1/2}$ the geopotential thickness is

$$\Delta\Phi = -R\int_{p_{k-1/2}}^{p_{k+1/2}} Td(ln p) = -R\sum T_i \ln\left(\frac{p_{iu}}{p_{ib}}\right) \qquad \textbf{(1)}$$

The model expression for the same thickness of a layer with mean potential temperature $\theta_k$ is

$$\Delta\Phi = -c_p\int_{p_{k-1/2}}^{p_{k+1/2}} \Theta d\Pi = -c_p\Theta_k\,(\Pi_{k+1/2} - \Pi_{k-1/2}) \qquad \textbf{(2)}$$

$$\Pi = (\frac{p}{p_0})^{\kappa}, \quad \kappa = \frac{R}{c_p} \qquad \textbf{(3)}$$

where the Exner pressure,

Matching (1) and (2) enables $\theta_k$ to be determined.

To convert to temperature at a model level, the Exner pressure at the "centre" of the layer is required. The model value for this is consistent with the geopotential equation and the dynamical formulation. It is given by

$$\Pi_k = \frac{\Delta(\Pi p)_k}{(\kappa+1)\ \Delta p_k} = \frac{\Pi_{k+1/2}p_{k+1/2} - \Pi_{k-1/2}p_{k-1/2}}{(\kappa+1)(p_{k+1/2} - p_{k-1/2})} \qquad \textbf{(4)}$$

so that

$$T_k = \Pi_k\ \Theta_k \qquad \textbf{(5)}$$

The original formulation of the Unified Model used the arithmetic mean of the layer boundary values for the Exner pressure at the level "centre", so:

$$\Pi_k^0 = \overline{(\Pi)}_k = 0.5\ (\Pi_{k=1/2} + \Pi_{k-1/2}) \qquad \textbf{(6)}$$

Both expressions (4) and (6) can give a warm bias to the derived temperatures. A simple test profile (Fig 4) was used as input and layer mean potential temperatures derived for the standard 19 layers (Table 1) using equations (1) and (2). Output temperatures at standard pressure levels show a warm bias increasing with altitude (Table 2), discounting the large errors associated with the tropopause.

Similar errors have been found in the operational global model as shown by the observed-model temperature differences for N. Atlantic sondes for November 1992 ( equation (4)) and December 1991 ( equation (6)).

b) New algorithm

Consider an isothermal layer of temperature T, with geopotential thickness given by

$$\Delta\Phi = -R\ T \ln\left(\frac{P_{k+1/2}}{p_{k-1/2}}\right) \qquad \textbf{(7)}$$

which implies a model layer potential temperature $\theta_k$ , by matching using equation (2) i.e.

$$-c_p \Theta_k \left(\Pi_{k+1/2} - \Pi_{k-1/2}\right) = -RT \ln\left(\frac{p_{k+1/2}}{p_{k-1/2}}\right) \qquad \textbf{(8)}$$

If there is to be no error in converting from $\theta_k$ to T at the nominal level, defined by the Exner value there, say $\Pi_m$, then

$$T = \Pi_m \Theta_k \qquad \textbf{(9)}$$

so that equation (8) implies that

$$-c_p \left(\Pi_{k+1/2} - \Pi_{k-1/2}\right) = -R\ \Pi_m \ln\left(\frac{p_{k+1/2}}{p_{k-1/2}}\right)$$

or

$$\boxed{\Pi_m = \frac{\Pi_{k+1/2} - \Pi_{k-1/2}}{\kappa\ (\ln p_{k+1/2} - \ln p_{k-1/2})} = \frac{\Delta\Pi_k}{\kappa\ \Delta(lnp)_k} \qquad \mathbf{(10)}}$$

This is the proposed new expression to be used for conversion from model layer potential temperatures to temperatures for output and derivation of background temperature profiles in the processing of locally retrieved satellite data. It should give smaller errors in the stratosphere where the atmosphere is close to isothermal. In the general case, it is evident that by its construction equation (10) will always result in the layer mean temperature,

$$\overline{T} = \frac{\int_{p_{k-1/2}}^{p_{k+1/2}} Td(\ln p)}{\int_{p_{k-1/2}}^{p_{k+1/2}} d(\ln p)} = \frac{\int_{p_{k-1/2}}^{p_{k+1/2}} Td(\ln p)}{\Delta(\ln p)_k} \qquad \mathbf{(11)}$$

being obtained from the layer mean $\theta_k$ . This temperature is assigned to a pressure, $p_m$ , from $\Pi_m = (p_m / p_0)$ . If temperature varies linearly with lnp across a model layer, (small) interpolation errors will result, since although the correct layer mean temperature is obtained, the nominal pressure does not equal the pressure (= $\sqrt{[p_{k+1/2}\ p_{k-1/2}]}$ ) where the mean temperature is attained.

A common practice in numerical models is to assign the nominal model level as located either by the arithmetic mean of the layer boundary pressures or the geometric mean of the layer boundary pressures; the latter is equivalent to finding the mean of the logarithm of the boundary pressures. The corresponding Exner pressure is given by

$$\Pi_{gm} = \sqrt{(\Pi_{k+1/2}\ \Pi_{k-1/2})} \qquad \mathbf{(12)}$$

This choice is most appropriate to models which use the T d(lnp) form of the geopotential equation with temperature as the model variable, and assuming that temperature varies linearly with lnp across a model layer. The relative location of these choices and those given by equations (4),(6) and the new proposal (10) are shown in Fig 5.

The test profile errors are much reduced with the new algorithm (Table 2) and are all less than 0.85K in magnitude, apart from the tropopause and the very top level, both of which cause problems to all the schemes due to the discontinuity and very thick top model layer. For comparison the errors using equation (12) are also shown; this always gives output temperatures which are too cold and are larger in magnitude than the new algorithm. The stratospheric errors are either more positive or negative depending on the relative locations of the nominal levels shown in Fig. 5.

In a global model test of the new algorithm in one 6-h assimilation the

Observation-background statistics (Table 3) show a clear reduction in bias from 300hPa upwards with slightly more observations passing the quality control. The new algorithm was also used in a recent trial of GLOSS. Verification results for sondes from the first day , which did not include any GLOSS data also show the expected reductions in bias at upper levels. For the north Atlantic and northern hemisphere regions the bias was almost reduced to zero , whilst in the Tropics and southern hemisphere there is now a smaller but negative bias . The southern hemisphere results should be treated with caution in view of the small number of radiosondes.

The geographical variation of the impact of changing to the new algorithm will depend on the time of year and temperature lapse rates which vary with season. Eg the differences found at a selection of standard pressure levels for one case, 00Z 1/02/93. The differences at 100hPa are of order 1K and increase upwards to differences of order 30-40K at 5hPa in agreement with the differences for the test profiles (Table 2, col3-col2).

## 4) Conclusions

The new algorithm clearly reduces the bias between observed temperatures and model output temperatures on standard pressure levels. It will also give a less biased profile for use in the radiation transfer calculations in the LASS/GLOSS system. It should therefore be implemented in all the operational versions of the unified model.

Further investigations should be done to assess whether changing to the new algorithm for all conversions between potential temperature and temperature within the model would be more accurate and beneficial. This will entail a redefinition of the model level constants.

All users of the unified model should be aware of the biases that may be due to the factors described here. Caution should be exercised when validating model temperatures, particularly at upper atmospheric levels when the model layer resolution is very coarse.

## Acknowledgements.

Colin Parrett obtained the global O-B statistics, and Alan Gadd provided verification results from the GLOSS trial.

TABLE 1

## STANDARD CONFIGURATIONS OF THE MODEL

The Climate, Global Forecast and Limited Area Forecast models use the following set of 19 levels.n=A/p +Bp , p =$10^5$ Pa. The layer boundaries are given by :

| Level | $A_{K+1/2}$ | $B_{K+1/2}$ | $\eta$ |
|---|---|---|---|
| 19.5 | 50.0 | 0.0 | 0.0005 |
| 18.5 | 1000.0 | 0.0 | 0.01 |
| 17.5 | 2000.0 | 0.0 | 0.02 |
| 16.5 | 4000.0 | 0.0 | 0.04 |
| 15.5 | 7176.0 | 0.003239 | 0.075 |
| 14.5 | 10652.1 | 0.018478 | 0.125 |
| 13.5 | 12997.5 | 0.045024 | 0.175 |
| 12.5 | 14342.7 | 0.081572 | 0.225 |
| 11.5 | 14818.3 | 0.126816 | 0.275 |
| 10.5 | 14555.1 | 0.179448 | 0.325 |
| 9.5 | 13447.6 | 0.250523 | 0.385 |
| 8.5 | 11175.3 | 0.348246 | 0.46 |
| 7.5 | 7727.9 | 0.472720 | 0.55 |
| 6.5 | 3852.2 | 0.611477 | 0.65 |
| 5.5 | 939.0 | 0.740609 | 0.75 |
| 4.5 | 0.0 | 0.835 | 0.835 |
| 3.5 | 0.0 | 0.905 | 0.905 |
| 2.5 | 0.0 | 0.956 | 0.956 |
| 1.5 | 0.0 | 0.994 | 0.994 |
| 0.5 | 0.0 | 1.0 | 1.0 |

Using the interpolation scheme defined following eq. (22) gives $A_k$, $B_k$ values:

| k | $A_k$ | $B_k$ | $\eta_k$ |
|---|---|---|---|
| 19 | 460.6 | 0.0 | 0.004606 |
| 18 | 1479.7 | 0.0 | 0.014797 |
| 17 | 2959.4 | 0.0 | 0.029594 |
| 16 | 5529.4 | 0.001560 | 0.056854 |
| 15 | 8861.7 | 0.010630 | 0.099247 |
| 14 | 11801.4 | 0.031487 | 0.149501 |
| 13 | 13660.1 | 0.063026 | 0.199627 |
| 12 | 14577.7 | 0.103925 | 0.249702 |
| 11 | 14688.1 | 0.152871 | 0.299752 |
| 10 | 14007.0 | 0.214628 | 0.354698 |
| 9 | 12323.5 | 0.298868 | 0.422103 |
| 8 | 9469.9 | 0.409823 | 0.504522 |
| 7 | 5809.3 | 0.541410 | 0.599503 |
| 6 | 2408.0 | 0.675494 | 0.699574 |
| 5 | 472.5 | 0.787503 | 0.792229 |
| 4 | 0.0 | 0.869832 | 0.869834 |
| 3 | 0.0 | 0.930417 | 0.930417 |
| 2 | 0.0 | 0.974956 | 0.974956 |
| 1 | 0.0 | 0.996999 | 0.996999 |

TABLE 2

Temperature interpolation errors ($T_{out}$- $T_{in}$ ) / K for test profile of Fig 1.

| pressure/ hPa | Exner pressure at nominal level | | | |
|---|---|---|---|---|
| | mean of boundary values, eq. 6 | model algorithm, eq. 4 | new algorithm,eq 10 | geometric mean, eq 12 |
| 1000 | -0.176 | -0.171 | -0.150 | -0.176 |
| 950 | 0.000 | 0.005 | -0.003 | -0.003 |
| 850 | -0.003 | 0.012 | -0.010 | -0.013 |
| 700 | 0.001 | 0.040 | -0.014 | -0.021 |
| 500 | -0.005 | 0.043 | -0.024 | -0.034 |
| 400 | -0.027 | 0.014 | -0.044 | -0.052 |
| 300 | -0.007 | 0.026 | -0.020 | -0.026 |
| 250 | -0.012 | 0.029 | -0.029 | -0.037 |
| 200 | 1.986 | 2.074 | 1.951 | 1.933 |
| 150 | 0.257 | 0.704 | 0.081 | -0.007 |
| 100 | 0.464 | 1.589 | 0.025 | -0.191 |
| 70 | 0.555 | 2.063 | -0.039 | -0.333 |
| 50 | 0.740 | 2.559 | 0.016 | -0.334 |
| 30 | 0.897 | 3.049 | 0.041 | -0.386 |
| 20 | 0.746 | 2.923 | -0.130 | -0.568 |
| 15 | 0.925 | 3.126 | 0.039 | -0.403 |
| 10 | 3.282 | 15.671 | -0.496 | -2.107 |
| 7 | 8.644 | 35.331 | 0.365 | -3.160 |
| 5 | 12.961 | 51.863 | 0.863 | -4.325 |
| 0.5 | 27.779 | 125.038 | -2.715 | -15.690 |

TABLE 3
Observation-Background temperature ( K) statistics for radiosondes, 00z 15/12/92 data

| pressure layer/ hPa | operational | | | new algorithm | | |
|---|---|---|---|---|---|---|
| | mean | rms | no passed q. control | mean | rms | no passed q.control |
| 1050-950 | 0.7 | 2.4 | 554 | 0.6 | 2.4 | 554 |
| 950-850 | 0.3 | 3.0 | 997 | 0.3 | 3.0 | 997 |
| 950-700 | 0.3 | 1.8 | 1123 | 0.4 | 2.0 | 1119 |
| 700-500 | 0.3 | 1.3 | 1496 | 0.3 | 1.3 | 1494 |
| 500-400 | 0.0 | 1.2 | 856 | 0.0 | 1.2 | 856 |
| 400-300 | -0.3 | 1.5 | 941 | -0.2 | 1.5 | 946 |
| 300-200 | -0.3 | 1.7 | 1119 | -0.2 | 1.7 | 1120 |
| 200-100 | -0.4 | 1.5 | 1209 | 0.0 | 1.5 | 1207 |
| 100-70 | -1.4 | 1.8 | 482 | -0.3 | 1.3 | 484 |
| 70-50 | -1.8 | 2.5 | 437 | 0.3 | 1.7 | 440 |
| 30-10 | -2.2 | 3.4 | 501 | 1.6 | 3.3 | 512 |
| all levels | -0.2 | 1.8 | 9715 | 0.2 | 1.8 | 9725 |

Figure 4

The variation is taken to be linear in ln(p) for the two lapse rates appropriate to an idealised troposphere and stratosphere.

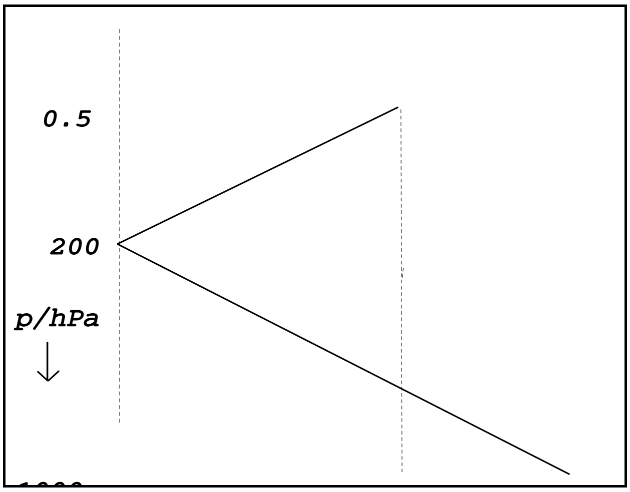

Figure 5
Relative locations of choices of nominal levels