交通部中央氣象局委託研究計畫成果報告

設計及研發全球伴隨波譜模式(I)

計畫類別 ： □國內 ☑國外
計畫編號 ： CWB 86 - 3M - 09
執行期間 ： 85 年 7 月 1 日至 86 年 6 月 30 日

計畫主持人 ： 鄒曉蕾
協同主持人 ： 張忍成

中華民國八十六年六月三十日

交通部中央氣象局委託
美國國家大氣研究中心
設計及研發全球伴隨波譜模式

美國國家大氣研究中心研究員：鄒曉蕾

中華民國八十六年六月

一、說明

伴隨理論(adjoint method)在氣象上的應用，最早可分為兩支(Daley, 1991)，以Marchuck (1974)和 Lions (1971)為代表， Marchuck 利用此法來探討氣候模式的敏感度(sensitivity)問題， Lions 則認為其不但能研究敏感度問題，對一個微分系統而言應該可以透過伴隨方法求算出最佳初始值。現今，這兩支理論已合而為一，其在數學分析中的推導僅有先後之別。換言之，在探討完敏感度後，接下來的挑戰就是：如何修正初始場而得到一個最佳的結果。由於氣象變數較多，方程組亦有相當的複雜性，故最初所遭遇的問題是：如何應用至氣象中？但，隨著數值方法和電腦硬體的進步，原先認為的龐大計算至今已不是問題；加上數值方法的進步，像早期的氣象學家如 Kontarev (1960)， LeDimet and Talagrand (1986)， Talagrand and Courtier (1987)等中所使用的方法，如今可用 Talagrand (1991)所提出的方法取代，對於伴隨算子有新的認知，模式的發展遠較從前容易；而隨後的 Errico and Vukicevic(1992)和 Zou(1995)等使用的方法都已經是用數值方法求算伴隨模式。

在許多氣象問題中，常需要求出一個泛函的極值，而所謂泛函即是由函數所組成的函數，例如，$\mathbf{x} \equiv (u, v, \Phi)^{\mathrm{T}}$,$\hat{\mathbf{x}} \equiv \left(\hat{u}, \hat{v}, \hat{\Phi}\right)^{\mathrm{T}}$ ，其中上標 T 表矩陣的轉置，欲求 $J \equiv (\mathbf{x} - \hat{\mathbf{x}})^2 / 2$ 的極值；此處 J 即為泛函。數學上，用變分學來求泛函的極值，而伴隨理論則為變分法中求泛函極值的標準計算步驟之一。

伴隨理論的精神，在於利用算子交換的特性。而為了要方便完成算子的交換，伴隨理論在線性理論的架構下探討問題，和原來線性方程有關的線性算子經過算子交換過程後得到伴隨算子，作用在另一變數上，其目的為規避對原方程變數做繁瑣的直接計算而能求得解答的一種方法。回溯數學歷史，伴隨算子分別在微分方程和線性代數中有相似的定義和不同的應用。微分方程中的伴隨算子，為應用在滿足特定邊界、初始條件而無法求得通解之情形下，找出特解的計算方法，常配合格林函

數(Green function)使用。在求泛函極值問題時，伴隨變數即為拉氏乘數(Lagrange multiplier)，其和極值的敏感度有關。而線性代數中的伴隨算子則用於坐標轉換的討論，求得在非正交坐標上，某一向量在各基底上的最佳投影。

為配和中央氣象局利用變分法發展四維資料同化之技術，本研究利用其全球波譜模式的動力部份為起點，開發此動力模式的伴隨模式，做為以後各項工作推展的基礎。本報告第二節簡述建立伴隨模式的原理與步驟，第三節則是討論運用伴隨模式時所應注意的事項和未來的工作計畫。

二、研究方法

現今的潮流是直接利用正切線性模式導出伴隨模式，此方法最早由 Talagrand (1991)提出，其優點是利於模式的發展和維護，本計畫所發展的伴隨模式將採行此法。正切線性模式為原非線性全球模式的一階導數近似，其數學上的推導如下：

原全球模式可簡寫成

$$\frac{\partial \mathbf{x}}{\partial t} = \mathbf{n}(\mathbf{x}) = \mathbf{F} \quad , \tag{1}$$

其中 $\mathbf{n}$ 是非線性算子，$\mathbf{x}$ 為變數所組成的行向量。設原變數 $\mathbf{x} = \bar{\mathbf{x}} + \mathbf{x}'$，其中 $\mathbf{x}'$ 為擾動場；而 $\bar{\mathbf{x}}$ 為一參考變數，可為不穩定度分析時之基本場，或是資料分析時之氣候場。將 $\mathbf{n}(\mathbf{x})$ 做 Taylor 級數展開，即

$$\mathbf{n}(\mathbf{x}) = \mathbf{n}(\bar{\mathbf{x}}) + \frac{\partial \mathbf{n}(\bar{\mathbf{x}})}{\partial \mathbf{x}}\mathbf{x}' + \mathbf{O}(\mathbf{x}'^2)$$

如忽略掉高次項，則(1)式可變為，

$$\frac{\partial \bar{\mathbf{x}}}{\partial t} + \frac{\partial \mathbf{x}'}{\partial t} = \mathbf{n}(\bar{\mathbf{x}}) + \frac{\partial \mathbf{n}(\bar{\mathbf{x}})}{\partial \mathbf{x}}\mathbf{x}' = \bar{\mathbf{F}} + \mathbf{f} \text{ 。}$$

令 $\frac{\partial \bar{\mathbf{x}}}{\partial t} = \mathbf{n}(\bar{\mathbf{x}}) = \bar{\mathbf{F}}$ ，

則此時，

$$\frac{\partial \mathbf{x}'}{\partial t} = \frac{\partial \mathbf{n}(\bar{\mathbf{x}})}{\partial \mathbf{x}}\mathbf{x}' = \mathbf{f} \text{ 。} \tag{2}$$

對 $\mathbf{x}'$ 言(2)為一線性方程。(2)式又稱為正切線性方程，和平常所認知的線性方程之不

同處，在於$\frac{\partial \mathbf{n}(\bar{\mathbf{x}})}{\partial \mathbf{x}}$這項來自(1)式的運算，為一時間的函數。於是上式可用線性算子 $\mathbf{L}$ 來取代$\frac{\partial \mathbf{n}(\bar{\mathbf{x}})}{\partial \mathbf{x}}$，則(2)式可簡寫成：

$$\frac{\partial \mathbf{x}'}{\partial t} \cong \dot{\mathbf{x}}' = \mathbf{L}\mathbf{x}' = \mathbf{f} \text{。}$$

上式僅是一則通式；換言之，全球模式中每一個變數或和此變數有關的相依變數都要依此做一階導數求得相應的正切線性方程。現以全球模式中之一項做範例，餘者可照此推導。

全球模式中的渦度方程可簡寫為：

$$\frac{\partial \zeta}{\partial t} = -\alpha(G, H) \text{，}$$

其中，

$$\begin{aligned} \alpha(G,H) &= \frac{1}{\cos^2\varphi}\frac{\partial G}{\partial \lambda} + \frac{\partial H}{\partial \mu} \\ G &= U(\zeta + f) + [\dot{\sigma}\pi]\left(\frac{\partial V}{\partial p}\right) + \frac{c_p}{a^2}\theta\left(\frac{\partial P}{\partial \pi}\right)\left(\frac{\partial \pi}{\partial \mu}\right)\cos^2\varphi - Q_v\frac{\cos\varphi}{a} \end{aligned} \tag{3}$$

而(3)式之正切線性方程為，

$$\begin{aligned} G' = {} & \overline{U}\zeta' + U'(\overline{\zeta} + f) + [\dot{\sigma}'\overline{\pi} + \overline{\dot{\sigma}}\pi']\left(\frac{\partial \overline{V}}{\partial \overline{p}}\right) + [\overline{\dot{\sigma}\pi}]\left(\frac{\partial V'\partial \overline{p} - \partial \overline{V}\partial p'}{(\partial \overline{p})^2}\right) \\ & + \frac{c_p}{a^2}\theta'\left(\frac{\partial \overline{P}}{\partial \overline{\pi}}\right)\left(\frac{\partial \overline{\pi}}{\partial \overline{\mu}}\right) + \frac{c_p}{a^2}\overline{\theta}\left(\frac{\partial P'\partial \overline{\pi} - \partial \overline{P}\partial \pi'}{(\partial \overline{\pi})^2}\right)\left(\frac{\partial \overline{\pi}}{\partial \overline{\mu}}\right) \\ & + \frac{c_p}{a^2}\overline{\theta}\left(\frac{\partial \overline{P}}{\partial \overline{\pi}}\right)\left(\frac{\partial \pi'\partial \overline{\mu} - \partial \overline{\pi}\partial \mu'}{(\partial \overline{\mu})^2}\right) - Q_v'\frac{\cos\varphi}{a} \end{aligned}$$

如此，經過逐次逐項的推導，可得到完整的正切線性方程。值得注意的是，所得之結果須滿足泰勒級數展開的一階近似，即

當 $\delta << 1$，則 $\dfrac{\mathbf{n}(\mathbf{x}+\delta\mathbf{x}')-\mathbf{n}(\mathbf{x})}{\delta\dfrac{\partial \mathbf{n}}{\partial \mathbf{x}}\mathbf{x}'} \cong 1$。

表一即為中央氣象局全球波譜正切線性模式的表現。須知上式不僅可對所有變數的整體表現做測試，亦可對單一變數做校驗。故表中對所有預報變數如 u, v, T, p_s, Q 等分別做檢驗，也對其合成的結果做檢查。結果顯示，此正切線性模式是正確的。

在完成正切線性模式之後接著便可以發展其伴隨模式。由於線性算子的特性是每一算式中只有一個變數，沒有變數在分母，也沒有變數的高次項，故獨立於此變數的所有部份皆可視為算子，可透過下述的推導，得此算子的轉置排列而得到伴隨模式。

於此，吾人用一簡單的平流方程 $\dot{u}' = \dfrac{\partial u'}{\partial t} = -\bar{u}\dfrac{\partial u'}{\partial x} - u'\dfrac{\partial \bar{u}}{\partial x}$ 來做範例，其寫成差分方程時成為，

$$\begin{pmatrix} \bullet \\ \bullet \\ \dot{u}'(i,j) \\ \bullet \\ \bullet \end{pmatrix} = \begin{pmatrix} \bullet\bullet & \bar{u}(i,j)\dfrac{1}{2\Delta x}, & -\dfrac{\bar{u}(i+1,j)-\bar{u}(i-1,j)}{2\Delta x}, & -\bar{u}(i,j)\dfrac{1}{2\Delta x} & \bullet\bullet \end{pmatrix} \begin{pmatrix} \bullet \\ \bullet \\ u'(i-1,j) \\ u'(i,j) \\ u'(i+1,j) \\ \bullet \\ \bullet \end{pmatrix}$$

$$= \mathbf{L}\mathbf{x} \qquad (4)$$

此時 $\mathbf{L}$ 為一線性矩陣算子，而 $\mathbf{x}$ 為一行向量變數，當對 $\mathbf{L}$ 取轉置時可得其伴隨算子 $\mathbf{L}^{\mathrm{T}}$ 和其對應之伴隨方程

$$\begin{pmatrix} \bullet \\ \bullet \\ \dot{u}'_a(i-1,j) \\ \dot{u}'_a(i,j) \\ \dot{u}'_a(i+1,j) \\ \bullet \\ \bullet \end{pmatrix} = \begin{pmatrix} \bullet \\ \bullet \\ \bar{u}(i,j)\dfrac{1}{2\Delta x}, \\ -\dfrac{\bar{u}(i+1,j)-\bar{u}(i-1,j)}{2\Delta x}, \\ -\bar{u}(i,j)\dfrac{1}{2\Delta x}, \\ \bullet \\ \bullet \end{pmatrix} \begin{pmatrix} \bullet \\ \bullet \\ \bullet \\ u'_a(i,j) \\ \bullet \\ \bullet \\ \bullet \end{pmatrix}$$

$$= \mathbf{L}^{\mathrm{T}}\mathbf{x}_{\mathbf{a}} \qquad (5)$$

比較(4)與(5)式，我們領悟到一個通則，就是把正切線性模式反寫回去即可得到伴隨模式。試看將(4)式等號左手邊的變數寫成伴隨變數，然後將其移至等號右手邊；隨後將原線性算子從新安排，即那一個變數所對應的線性算子將仍然是其相應伴隨變數的線性算子，但為了排成矩陣的型式起見，它必須旋轉 90 度，亦即原矩陣的轉置；最後將原等號右手邊的變數移至左手邊。這就是 Talagrand 在 1991 年所提出的方法，它大大節省了發展伴隨模式的時間，更重要的是它和原來的正切線性模式有相互的依存關係，並不需要給定特別的邊界條件，這就是後來稱為數值模式的伴隨模式之由來，因其並非從原來的方程透過解析運算而得到的伴隨模式。表二即為利用此一精神所撰寫的簡單範例，該演算法(algorithm)的精神為全球模式中所用的調整性蛙跳積分格式(filtered leap-frog time integration scheme)。利用此法即可逐條修改每一個正切線性模式的副程式而得到一完整的伴隨模式。

三、結論與未來方向

所得出的伴隨模式必須通過下面伴隨恆等式之的檢測，即

$$\langle \mathbf{x}_a'(t)\ ,\ \mathbf{x}'(t) \rangle = \langle \mathbf{x}_a'(t)\ ,\ \mathbf{L}(t)\,\mathbf{x}'(t_0) \rangle$$
$$= \langle \mathbf{L}^{\mathrm{T}}(t)\,\mathbf{x}_a'(t)\ ,\ \mathbf{x}'(t_0) \rangle = \langle \mathbf{x}_a'(t_0)\ ,\ \mathbf{x}'(t_0) \rangle$$

其中 **L** 為正切線性算子，$\mathbf{x}'$ 為相應於原模式之外的一小擾動變數，$\mathbf{x}_a'$ 為其伴隨變數，$\mathbf{L}^{\mathrm{T}}$ 為伴隨算子，t 表時間，$\langle\ ,\ \rangle$ 括弧表函數空間的內積。上式意為當於某一時刻之伴隨變數和小擾動變數的內積，恆等於初始狀態時伴隨變數和小擾動變數的內積。於 64 位元的電腦，如 Cray-YMP 上應該有 10 個數值位的精確度。

總計所修改的正切線性模式和伴隨模式如表三中所示，所有的程式附於本報告之末以為參考。其取名的方式按照原先非線性模式的名稱但在其前加一個大寫的 M 表示，加 L 代表為正切線性模式，而加 adL 表為伴隨模式。

伴隨理論源自於線性代數和微分方程算子，其後應用於變分最佳化分析。由於求算伴隨模式數值方法的進步，可消弭線性代數和微分方程中伴隨算子恆等式的差異，無須事先預設所求問題之邊界條件來推導伴隨方程。據此求出之伴隨變數，正是所定義之價值函數(cost function)以初始場變數來表示之梯度，而依據此梯度值可用數值疊代法求解當初始場為若干時，其價值函數的極值。

不同的價值函數定義，將定義出不同的伴隨變數，而其代表對所求問題之敏感性。伴隨變數值愈大的區域，表示這塊地區對所求問題的影響很大；任何一個存於此區的小擾動將影響後來的結果甚鉅，需要比其它地區多做調整。但非常重要的是：真實世界的變化，不一定要照著伴隨變數所指示的數量級做調整才能得到所要的結果；也就是說，伴隨理論提供的是一最佳化的方法，意即如何調整初始場最少，而能求得最接進所設想的狀態。

舉凡在氣象應用中牽涉到變分求極值的問題都可藉助伴隨理論求解。因為任何一種求解過程都可視為由一運算子和其變數所組成的函數，只要能將運算子線性化，都能得到伴隨算子而求解極值。但是，原理容易了解，實際作業時方程組的複雜性、，電腦計算資源的大小都可能影響到伴隨模式的實用性，但儘管如此，由於其可明確計算出四維空間變數的梯度值，在處理變分極值問題時仍有很大潛力。此外，對不易直接診斷的模式敏感度測試亦有很大幫助；而應用在不穩定度和可預測度的問題上，亦是不錯的工具。

運用前述的數值法推導伴隨模式似乎是比較好的選擇，可以藉由計算伴隨算子的恆等式是否相等來驗證伴隨模式是否正確，另一方面是在模式的發展維護上，較為容易。此外，必須要特別注意的是，伴隨模式並非是一個時間逆向的模式而已，它所得出的值是一個梯度，一個敏感度區，決非當定義伴隨變數初始值為 $\mathbf{x}'_a(t_f) \equiv \mathbf{x}(t_f) - \bar{\mathbf{x}}(t_f)$，其積分終了時所得到的變數是 $\mathbf{x}'_a(t_0) = \mathbf{x}(t_0) - \bar{\mathbf{x}}(t_0)$，這是一個嚴重的認知錯誤。

至於未來的工作則分為下列幾點：

(一)利用伴隨模式做極小化程序(minimization)的測試，這是做為四維資料同化技術的第一步工作。

(二)作敏感度測試以驗証並校驗中央氣象局的全球模式。

(三)加入幾個簡單的物理過程，如大尺度降水和簡單的幅射過程，使此伴隨模式更加完整，更符合實際大氣的調整變化，以應付後續實際作業的需要。我們將在後續的合作計畫中，逐項進行這個工作。

四、參考文獻

Courtier, P., J. Derber, R. M. Errico, J.-F. Louis and T. Vukicevic, 1993 : Important literature on the use of adjoint, variational methods and the Kalman filter in Meterorology. *Tellus*, **45A**, 257-342.

Daley, R., 1991: *Atmospheric data analysis*. Cambridge University press.

Errico, R. M. and T. Vukicevic, 1992 : Sensitivity analysis using an adjoint of PSU-NCAR mesoscale model. *Mon. Wea. Rev.*, **120**, 1644-1660.

Kontarev, G.,1980 : The adjoint equation technique applied to meteorological problems. *ECMWF Tech. Rep.,No.* **21**, Shinfield Park, Reading, RG29AX,U.K., 21pp.

LeDimet, F. X., and O. Talagrand, 1986 : Variational algorithms for analysis and assimilation of meteorological observations: Theoretical aspects. *Tellus*, **38A**, 97-110.

Navon, I. M., X. Zou, J. Derber, and J, Sela, 1992 : Variational data assimilation with an adiabtic version of the NMC spectral model. *Mon. Wea. Rev.*, **120**, 1433-1466.

----------, and D. M. Legler, 1987 : Conjugated-gradient methods for large scale minimization in meteorology. *Mon. Wea. Rev.*, **115**, 1479-1502.

Talagrand, O., and P. Courtier, 1987 : Variational assimilation of Meterorological observations with adjoint vorticity equation. Part I. Theory. *Quart. J. Roy. Meteor. Soc.*, **113**, 1311-1328.

Talagrand, O., 1991 : The use of adjoint equations in numerical modeling of the atmospheric circulation. In: (A. Griewank,G. Corlise,editor) *Automatic differentiation of algorithms:theory,implementation and application*,pp. 169-180.

Zou, X., Y.-H. Kuo, and Y.-R. Guo, 1995: Assimilation of atmospheric radio refractivity using a nonhydrostatic adjoint model. *Mon. Wea. Rev.*, **123**, 2229-2249.

--------, I.M. Navon, and F. X. Le Dimet, 1992 : Incomplete observations and control of gravity waves in variational data assimilation. *Tellus*, **44A**, 272-296.

表一

User gfsll@cray2e Jun 19 10:14 1997 tlmcheck.dat Page 1

```
afa= 1.0000000000    Taylor test=      1.00000007425401
afa= 1.0000000000    k=  1   U comp. Taylor test=      0.9992289317759
afa= 1.0000000000    k=  2   U comp. Taylor test=      0.9995608629394
afa= 1.0000000000    k=  3   U comp. Taylor test=      0.9995608629394
afa= 1.0000000000    k=  4   U comp. Taylor test=      0.9995608629394
afa= 1.0000000000    k=  5   U comp. Taylor test=      0.9995608629394
afa= 1.0000000000    k=  6   U comp. Taylor test=      0.9995608629394
afa= 1.0000000000    k=  7   U comp. Taylor test=      0.9995608629394
afa= 1.0000000000    k=  8   U comp. Taylor test=      0.9995608629394
afa= 1.0000000000    k=  9   U comp. Taylor test=      0.9995608629394
afa= 1.0000000000    k= 10   U comp. Taylor test=      0.9995608629394
afa= 1.0000000000    k= 11   U comp. Taylor test=      0.9995608629394
afa= 1.0000000000    k= 12   U comp. Taylor test=      0.9995608629394
afa= 1.0000000000    k= 13   U comp. Taylor test=      0.9995608629394
afa= 1.0000000000    k= 14   U comp. Taylor test=      0.9995608629394
afa= 1.0000000000    k= 15   U comp. Taylor test=      0.9995608629394
afa= 1.0000000000    k= 16   U comp. Taylor test=      0.9995608629394
afa= 1.0000000000    k= 17   U comp. Taylor test=      0.9995608629394
afa= 1.0000000000    k= 18   U comp. Taylor test=      0.9995608629394
afa= 1.0000000000    k=  1   V comp. Taylor test=      0.9990445504431
afa= 1.0000000000    k=  2   V comp. Taylor test=      0.9990303138954
afa= 1.0000000000    k=  3   V comp. Taylor test=      0.9990303138954
afa= 1.0000000000    k=  4   V comp. Taylor test=      0.9990303138954
afa= 1.0000000000    k=  5   V comp. Taylor test=      0.9990303138954
afa= 1.0000000000    k=  6   V comp. Taylor test=      0.9990303138954
afa= 1.0000000000    k=  7   V comp. Taylor test=      0.9990303138954
afa= 1.0000000000    k=  8   V comp. Taylor test=      0.9990303138954
afa= 1.0000000000    k=  9   V comp. Taylor test=      0.9990303138954
afa= 1.0000000000    k= 10   V comp. Taylor test=      0.9990303138954
afa= 1.0000000000    k= 11   V comp. Taylor test=      0.9990303138954
afa= 1.0000000000    k= 12   V comp. Taylor test=      0.9990303138954
afa= 1.0000000000    k= 13   V comp. Taylor test=      0.9990303138954
afa= 1.0000000000    k= 14   V comp. Taylor test=      0.9990303138954
afa= 1.0000000000    k= 15   V comp. Taylor test=      0.9990303138954
afa= 1.0000000000    k= 16   V comp. Taylor test=      0.9990303138954
afa= 1.0000000000    k= 17   V comp. Taylor test=      0.9990303138954
afa= 1.0000000000    k= 18   V comp. Taylor test=      0.9990303138954
afa= 1.0000000000    k=  1   T comp. Taylor test=      1.0000113491048
afa= 1.0000000000    k=  2   T comp. Taylor test=      1.0000076513493
afa= 1.0000000000    k=  3   T comp. Taylor test=      1.0000076513493
afa= 1.0000000000    k=  4   T comp. Taylor test=      1.0000076513493
afa= 1.0000000000    k=  5   T comp. Taylor test=      1.0000076513493
afa= 1.0000000000    k=  6   T comp. Taylor test=      1.0000076513493
afa= 1.0000000000    k=  7   T comp. Taylor test=      1.0000076513493
afa= 1.0000000000    k=  8   T comp. Taylor test=      1.0000076513493
afa= 1.0000000000    k=  9   T comp. Taylor test=      1.0000076513493
afa= 1.0000000000    k= 10   T comp. Taylor test=      1.0000076513493
afa= 1.0000000000    k= 11   T comp. Taylor test=      1.0000076513493
afa= 1.0000000000    k= 12   T comp. Taylor test=      1.0000076513493
afa= 1.0000000000    k= 13   T comp. Taylor test=      1.0000076513493
afa= 1.0000000000    k= 14   T comp. Taylor test=      1.0000076513493
afa= 1.0000000000    k= 15   T comp. Taylor test=      1.0000076513493
afa= 1.0000000000    k= 16   T comp. Taylor test=      1.0000076513493
afa= 1.0000000000    k= 17   T comp. Taylor test=      1.0000076513493
afa= 1.0000000000    k= 18   T comp. Taylor test=      1.0000076513493
afa= 1.0000000000    k=  1   Q comp. Taylor test=      1.0002582350781
```

User gfsll@cray2e Jun 19 10:14 1997 tlmcheck.dat Page 2

```
afa= 1.0000000000    k=  2   Q comp. Taylor test=      1.0002582350781
afa= 1.0000000000    k=  3   Q comp. Taylor test=      1.0002582350781
afa= 1.0000000000    k=  4   Q comp. Taylor test=      1.0002582350781
afa= 1.0000000000    k=  5   Q comp. Taylor test=      1.0002582350781
afa= 1.0000000000    k=  6   Q comp. Taylor test=      1.0002582350781
afa= 1.0000000000    k=  7   Q comp. Taylor test=      1.0002582350781
afa= 1.0000000000    k=  8   Q comp. Taylor test=      1.0002582350781
afa= 1.0000000000    k=  9   Q comp. Taylor test=      1.0002582350781
afa= 1.0000000000    k= 10   Q comp. Taylor test=      1.0002582350781
afa= 1.0000000000    k= 11   Q comp. Taylor test=      1.0002582350781
afa= 1.0000000000    k= 12   Q comp. Taylor test=      1.0002582350781
afa= 1.0000000000    k= 13   Q comp. Taylor test=      1.0002582350781
afa= 1.0000000000    k= 14   Q comp. Taylor test=      1.0002582350781
afa= 1.0000000000    k= 15   Q comp. Taylor test=      1.0002582350781
afa= 1.0000000000    k= 16   Q comp. Taylor test=      1.0002582350781
afa= 1.0000000000    k= 17   Q comp. Taylor test=      1.0002582350781
afa= 1.0000000000    k= 18   Q comp. Taylor test=      1.0002582350781
afa= 1.0000000000    Ps comp. Taylor test=      0.9999071154139
afa= 0.1000000000    Taylor test=      1.0000001901287
afa= 0.1000000000    k=  1   U comp. Taylor test=      1.0002992104424
afa= 0.1000000000    k=  2   U comp. Taylor test=      1.0003715285476
afa= 0.1000000000    k=  3   U comp. Taylor test=      1.0003715285476
afa= 0.1000000000    k=  4   U comp. Taylor test=      1.0003715285476
afa= 0.1000000000    k=  5   U comp. Taylor test=      1.0003715285476
afa= 0.1000000000    k=  6   U comp. Taylor test=      1.0003715285476
afa= 0.1000000000    k=  7   U comp. Taylor test=      1.0003715285476
afa= 0.1000000000    k=  8   U comp. Taylor test=      1.0003715285476
afa= 0.1000000000    k=  9   U comp. Taylor test=      1.0003715285476
afa= 0.1000000000    k= 10   U comp. Taylor test=      1.0003715285476
afa= 0.1000000000    k= 11   U comp. Taylor test=      1.0003715285476
afa= 0.1000000000    k= 12   U comp. Taylor test=      1.0003715285476
afa= 0.1000000000    k= 13   U comp. Taylor test=      1.0003715285476
afa= 0.1000000000    k= 14   U comp. Taylor test=      1.0003715285476
afa= 0.1000000000    k= 15   U comp. Taylor test=      1.0003715285476
afa= 0.1000000000    k= 16   U comp. Taylor test=      1.0003715285476
afa= 0.1000000000    k= 17   U comp. Taylor test=      1.0003715285476
afa= 0.1000000000    k= 18   U comp. Taylor test=      1.0003715285476
afa= 0.1000000000    k=  1   V comp. Taylor test=      0.9996854087844
afa= 0.1000000000    k=  2   V comp. Taylor test=      0.9996841531587
afa= 0.1000000000    k=  3   V comp. Taylor test=      0.9996841531587
afa= 0.1000000000    k=  4   V comp. Taylor test=      0.9996841531587
afa= 0.1000000000    k=  5   V comp. Taylor test=      0.9996841531587
afa= 0.1000000000    k=  6   V comp. Taylor test=      0.9996841531587
afa= 0.1000000000    k=  7   V comp. Taylor test=      0.9996841531587
afa= 0.1000000000    k=  8   V comp. Taylor test=      0.9996841531587
afa= 0.1000000000    k=  9   V comp. Taylor test=      0.9996841531587
afa= 0.1000000000    k= 10   V comp. Taylor test=      0.9996841531587
afa= 0.1000000000    k= 11   V comp. Taylor test=      0.9996841531587
afa= 0.1000000000    k= 12   V comp. Taylor test=      0.9996841531587
afa= 0.1000000000    k= 13   V comp. Taylor test=      0.9996841531587
afa= 0.1000000000    k= 14   V comp. Taylor test=      0.9996841531587
afa= 0.1000000000    k= 15   V comp. Taylor test=      0.9996841531587
afa= 0.1000000000    k= 16   V comp. Taylor test=      0.9996841531587
afa= 0.1000000000    k= 17   V comp. Taylor test=      0.9996841531587
afa= 0.1000000000    k= 18   V comp. Taylor test=      0.9996841531587
afa= 0.1000000000    k=  1   T comp. Taylor test=      1.0000008715060
```

```
 afa= 0.1000000000    k=  2   T comp. Taylor test=      1.0000008148154
 afa= 0.1000000000    k=  3   T comp. Taylor test=      1.0000008148154
 afa= 0.1000000000    k=  4   T comp. Taylor test=      1.0000008148154
 afa= 0.1000000000    k=  5   T comp. Taylor test=      1.0000008148154
 afa= 0.1000000000    k=  6   T comp. Taylor test=      1.0000008148154
 afa= 0.1000000000    k=  7   T comp. Taylor test=      1.0000008148154
 afa= 0.1000000000    k=  8   T comp. Taylor test=      1.0000008148154
 afa= 0.1000000000    k=  9   T comp. Taylor test=      1.0000008148154
 afa= 0.1000000000    k= 10   T comp. Taylor test=      1.0000008148154
 afa= 0.1000000000    k= 11   T comp. Taylor test=      1.0000008148154
 afa= 0.1000000000    k= 12   T comp. Taylor test=      1.0000008148154
 afa= 0.1000000000    k= 13   T comp. Taylor test=      1.0000008148154
 afa= 0.1000000000    k= 14   T comp. Taylor test=      1.0000008148154
 afa= 0.1000000000    k= 15   T comp. Taylor test=      1.0000008148154
 afa= 0.1000000000    k= 16   T comp. Taylor test=      1.0000008148154
 afa= 0.1000000000    k= 17   T comp. Taylor test=      1.0000008148154
 afa= 0.1000000000    k= 18   T comp. Taylor test=      1.0000008148154
 afa= 0.1000000000    k=  1   Q comp. Taylor test=      1.0000229193351
 afa= 0.1000000000    k=  2   Q comp. Taylor test=      1.0000229193351
 afa= 0.1000000000    k=  3   Q comp. Taylor test=      1.0000229193351
 afa= 0.1000000000    k=  4   Q comp. Taylor test=      1.0000229193351
 afa= 0.1000000000    k=  5   Q comp. Taylor test=      1.0000229193351
 afa= 0.1000000000    k=  6   Q comp. Taylor test=      1.0000229193351
 afa= 0.1000000000    k=  7   Q comp. Taylor test=      1.0000229193351
 afa= 0.1000000000    k=  8   Q comp. Taylor test=      1.0000229193351
 afa= 0.1000000000    k=  9   Q comp. Taylor test=      1.0000229193351
 afa= 0.1000000000    k= 10   Q comp. Taylor test=      1.0000229193351
 afa= 0.1000000000    k= 11   Q comp. Taylor test=      1.0000229193351
 afa= 0.1000000000    k= 12   Q comp. Taylor test=      1.0000229193351
 afa= 0.1000000000    k= 13   Q comp. Taylor test=      1.0000229193351
 afa= 0.1000000000    k= 14   Q comp. Taylor test=      1.0000229193351
 afa= 0.1000000000    k= 15   Q comp. Taylor test=      1.0000229193351
 afa= 0.1000000000    k= 16   Q comp. Taylor test=      1.0000229193351
 afa= 0.1000000000    k= 17   Q comp. Taylor test=      1.0000229193351
 afa= 0.1000000000    k= 18   Q comp. Taylor test=      1.0000229193351
 afa= 0.1000000000    Ps comp. Taylor test=      0.9999917248561
 afa= 0.0100000000    Taylor test=      1.0000001365859
 afa= 0.0100000000    k=  1   U comp. Taylor test=      1.0004061953648
 afa= 0.0100000000    k=  2   U comp. Taylor test=      1.0004521605256
 afa= 0.0100000000    k=  3   U comp. Taylor test=      1.0004521605256
 afa= 0.0100000000    k=  4   U comp. Taylor test=      1.0004521605256
 afa= 0.0100000000    k=  5   U comp. Taylor test=      1.0004521605256
 afa= 0.0100000000    k=  6   U comp. Taylor test=      1.0004521605256
 afa= 0.0100000000    k=  7   U comp. Taylor test=      1.0004521605256
 afa= 0.0100000000    k=  8   U comp. Taylor test=      1.0004521605256
 afa= 0.0100000000    k=  9   U comp. Taylor test=      1.0004521605256
 afa= 0.0100000000    k= 10   U comp. Taylor test=      1.0004521605256
 afa= 0.0100000000    k= 11   U comp. Taylor test=      1.0004521605256
 afa= 0.0100000000    k= 12   U comp. Taylor test=      1.0004521605256
 afa= 0.0100000000    k= 13   U comp. Taylor test=      1.0004521605256
 afa= 0.0100000000    k= 14   U comp. Taylor test=      1.0004521605256
 afa= 0.0100000000    k= 15   U comp. Taylor test=      1.0004521605256
 afa= 0.0100000000    k= 16   U comp. Taylor test=      1.0004521605256
 afa= 0.0100000000    k= 17   U comp. Taylor test=      1.0004521605256
 afa= 0.0100000000    k= 18   U comp. Taylor test=      1.0004521605256
```



```
 afa= 0.0100000000    k=  2   V comp. Taylor test=      0.9997497327228
 afa= 0.0100000000    k=  3   V comp. Taylor test=      0.9997497327228
 afa= 0.0100000000    k=  4   V comp. Taylor test=      0.9997497327228
 afa= 0.0100000000    k=  5   V comp. Taylor test=      0.9997497327228
 afa= 0.0100000000    k=  6   V comp. Taylor test=      0.9997497327228
 afa= 0.0100000000    k=  7   V comp. Taylor test=      0.9997497327228
 afa= 0.0100000000    k=  8   V comp. Taylor test=      0.9997497327228
 afa= 0.0100000000    k=  9   V comp. Taylor test=      0.9997497327228
 afa= 0.0100000000    k= 10   V comp. Taylor test=      0.9997497327228
 afa= 0.0100000000    k= 11   V comp. Taylor test=      0.9997497327228
 afa= 0.0100000000    k= 12   V comp. Taylor test=      0.9997497327228
 afa= 0.0100000000    k= 13   V comp. Taylor test=      0.9997497327228
 afa= 0.0100000000    k= 14   V comp. Taylor test=      0.9997497327228
 afa= 0.0100000000    k= 15   V comp. Taylor test=      0.9997497327228
 afa= 0.0100000000    k= 16   V comp. Taylor test=      0.9997497327228
 afa= 0.0100000000    k= 17   V comp. Taylor test=      0.9997497327228
 afa= 0.0100000000    k= 18   V comp. Taylor test=      0.9997497327228
 afa= 0.0100000000    k=  1   T comp. Taylor test=      0.9999998281361
 afa= 0.0100000000    k=  2   T comp. Taylor test=      1.0000001342738
 afa= 0.0100000000    k=  3   T comp. Taylor test=      1.0000001342738
 afa= 0.0100000000    k=  4   T comp. Taylor test=      1.0000001342738
 afa= 0.0100000000    k=  5   T comp. Taylor test=      1.0000001342738
 afa= 0.0100000000    k=  6   T comp. Taylor test=      1.0000001342738
 afa= 0.0100000000    k=  7   T comp. Taylor test=      1.0000001342738
 afa= 0.0100000000    k=  8   T comp. Taylor test=      1.0000001342738
 afa= 0.0100000000    k=  9   T comp. Taylor test=      1.0000001342738
 afa= 0.0100000000    k= 10   T comp. Taylor test=      1.0000001342738
 afa= 0.0100000000    k= 11   T comp. Taylor test=      1.0000001342738
 afa= 0.0100000000    k= 12   T comp. Taylor test=      1.0000001342738
 afa= 0.0100000000    k= 13   T comp. Taylor test=      1.0000001342738
 afa= 0.0100000000    k= 14   T comp. Taylor test=      1.0000001342738
 afa= 0.0100000000    k= 15   T comp. Taylor test=      1.0000001342738
 afa= 0.0100000000    k= 16   T comp. Taylor test=      1.0000001342738
 afa= 0.0100000000    k= 17   T comp. Taylor test=      1.0000001342738
 afa= 0.0100000000    k= 18   T comp. Taylor test=      1.0000001342738
 afa= 0.0100000000    k=  1   Q comp. Taylor test=      0.9999994330594
 afa= 0.0100000000    k=  2   Q comp. Taylor test=      0.9999994330594
 afa= 0.0100000000    k=  3   Q comp. Taylor test=      0.9999994330594
 afa= 0.0100000000    k=  4   Q comp. Taylor test=      0.9999994330594
 afa= 0.0100000000    k=  5   Q comp. Taylor test=      0.9999994330594
 afa= 0.0100000000    k=  6   Q comp. Taylor test=      0.9999994330594
 afa= 0.0100000000    k=  7   Q comp. Taylor test=      0.9999994330594
 afa= 0.0100000000    k=  8   Q comp. Taylor test=      0.9999994330594
 afa= 0.0100000000    k=  9   Q comp. Taylor test=      0.9999994330594
 afa= 0.0100000000    k= 10   Q comp. Taylor test=      0.9999994330594
 afa= 0.0100000000    k= 11   Q comp. Taylor test=      0.9999994330594
 afa= 0.0100000000    k= 12   Q comp. Taylor test=      0.9999994330594
 afa= 0.0100000000    k= 13   Q comp. Taylor test=      0.9999994330594
 afa= 0.0100000000    k= 14   Q comp. Taylor test=      0.9999994330594
 afa= 0.0100000000    k= 15   Q comp. Taylor test=      0.9999994330594
 afa= 0.0100000000    k= 16   Q comp. Taylor test=      0.9999994330594
 afa= 0.0100000000    k= 17   Q comp. Taylor test=      0.9999994330594
 afa= 0.0100000000    k= 18   Q comp. Taylor test=      0.9999994330594
 afa= 0.0100000000    Ps comp. Taylor test=      1.0000001679180
 afa= 0.0010000000    Taylor test=      1.0000001306009
```

```
afa= 0.0010000000    k=  2   U comp. Taylor test=      1.0004606073905
afa= 0.0010000000    k=  3   U comp. Taylor test=      1.0004606073905
afa= 0.0010000000    k=  4   U comp. Taylor test=      1.0004606073905
afa= 0.0010000000    k=  5   U comp. Taylor test=      1.0004606073905
afa= 0.0010000000    k=  6   U comp. Taylor test=      1.0004606073905
afa= 0.0010000000    k=  7   U comp. Taylor test=      1.0004606073905
afa= 0.0010000000    k=  8   U comp. Taylor test=      1.0004606073905
afa= 0.0010000000    k=  9   U comp. Taylor test=      1.0004606073905
afa= 0.0010000000    k= 10   U comp. Taylor test=      1.0004606073905
afa= 0.0010000000    k= 11   U comp. Taylor test=      1.0004606073905
afa= 0.0010000000    k= 12   U comp. Taylor test=      1.0004606073905
afa= 0.0010000000    k= 13   U comp. Taylor test=      1.0004606073905
afa= 0.0010000000    k= 14   U comp. Taylor test=      1.0004606073905
afa= 0.0010000000    k= 15   U comp. Taylor test=      1.0004606073905
afa= 0.0010000000    k= 16   U comp. Taylor test=      1.0004606073905
afa= 0.0010000000    k= 17   U comp. Taylor test=      1.0004606073905
afa= 0.0010000000    k= 18   U comp. Taylor test=      1.0004606073905
afa= 0.0010000000    k=  1   V comp. Taylor test=      0.9997559350719
afa= 0.0010000000    k=  2   V comp. Taylor test=      0.9997562165069
afa= 0.0010000000    k=  3   V comp. Taylor test=      0.9997562165069
afa= 0.0010000000    k=  4   V comp. Taylor test=      0.9997562165069
afa= 0.0010000000    k=  5   V comp. Taylor test=      0.9997562165069
afa= 0.0010000000    k=  6   V comp. Taylor test=      0.9997562165069
afa= 0.0010000000    k=  7   V comp. Taylor test=      0.9997562165069
afa= 0.0010000000    k=  8   V comp. Taylor test=      0.9997562165069
afa= 0.0010000000    k=  9   V comp. Taylor test=      0.9997562165069
afa= 0.0010000000    k= 10   V comp. Taylor test=      0.9997562165069
afa= 0.0010000000    k= 11   V comp. Taylor test=      0.9997562165069
afa= 0.0010000000    k= 12   V comp. Taylor test=      0.9997562165069
afa= 0.0010000000    k= 13   V comp. Taylor test=      0.9997562165069
afa= 0.0010000000    k= 14   V comp. Taylor test=      0.9997562165069
afa= 0.0010000000    k= 15   V comp. Taylor test=      0.9997562165069
afa= 0.0010000000    k= 16   V comp. Taylor test=      0.9997562165069
afa= 0.0010000000    k= 17   V comp. Taylor test=      0.9997562165069
afa= 0.0010000000    k= 18   V comp. Taylor test=      0.9997562165069
afa= 0.0010000000    k=  1   T comp. Taylor test=      0.9999997219535
afa= 0.0010000000    k=  2   T comp. Taylor test=      1.0000000651689
afa= 0.0010000000    k=  3   T comp. Taylor test=      1.0000000651689
afa= 0.0010000000    k=  4   T comp. Taylor test=      1.0000000651689
afa= 0.0010000000    k=  5   T comp. Taylor test=      1.0000000651689
afa= 0.0010000000    k=  6   T comp. Taylor test=      1.0000000651689
afa= 0.0010000000    k=  7   T comp. Taylor test=      1.0000000651689
afa= 0.0010000000    k=  8   T comp. Taylor test=      1.0000000651689
afa= 0.0010000000    k=  9   T comp. Taylor test=      1.0000000651689
afa= 0.0010000000    k= 10   T comp. Taylor test=      1.0000000651689
afa= 0.0010000000    k= 11   T comp. Taylor test=      1.0000000651689
afa= 0.0010000000    k= 12   T comp. Taylor test=      1.0000000651689
afa= 0.0010000000    k= 13   T comp. Taylor test=      1.0000000651689
afa= 0.0010000000    k= 14   T comp. Taylor test=      1.0000000651689
afa= 0.0010000000    k= 15   T comp. Taylor test=      1.0000000651689
afa= 0.0010000000    k= 16   T comp. Taylor test=      1.0000000651689
afa= 0.0010000000    k= 17   T comp. Taylor test=      1.0000000651689
afa= 0.0010000000    k= 18   T comp. Taylor test=      1.0000000651689
afa= 0.0010000000    k=  1   Q comp. Taylor test=      0.9999970625528
afa= 0.0010000000    k=  2   Q comp. Taylor test=      0.9999970625528
afa= 0.0010000000    k=  3   Q comp. Taylor test=      0.9999970625528
```



```
afa= 0.0010000000    k=  4   Q comp. Taylor test=      0.9999970625528
afa= 0.0010000000    k=  5   Q comp. Taylor test=      0.9999970625528
afa= 0.0010000000    k=  6   Q comp. Taylor test=      0.9999970625528
afa= 0.0010000000    k=  7   Q comp. Taylor test=      0.9999970625528
afa= 0.0010000000    k=  8   Q comp. Taylor test=      0.9999970625528
afa= 0.0010000000    k=  9   Q comp. Taylor test=      0.9999970625528
afa= 0.0010000000    k= 10   Q comp. Taylor test=      0.9999970625528
afa= 0.0010000000    k= 11   Q comp. Taylor test=      0.9999970625528
afa= 0.0010000000    k= 12   Q comp. Taylor test=      0.9999970625528
afa= 0.0010000000    k= 13   Q comp. Taylor test=      0.9999970625528
afa= 0.0010000000    k= 14   Q comp. Taylor test=      0.9999970625528
afa= 0.0010000000    k= 15   Q comp. Taylor test=      0.9999970625528
afa= 0.0010000000    k= 16   Q comp. Taylor test=      0.9999970625528
afa= 0.0010000000    k= 17   Q comp. Taylor test=      0.9999970625528
afa= 0.0010000000    k= 18   Q comp. Taylor test=      0.9999970625528
afa= 0.0010000000    Ps comp. Taylor test=      1.0000010172806
afa= 0.0001000000    Taylor test=      1.0000001181017
afa= 0.0001000000    k=  1   U comp. Taylor test=      1.0004251912132
afa= 0.0001000000    k=  2   U comp. Taylor test=      1.0004670166177
afa= 0.0001000000    k=  3   U comp. Taylor test=      1.0004670166177
afa= 0.0001000000    k=  4   U comp. Taylor test=      1.0004670166177
afa= 0.0001000000    k=  5   U comp. Taylor test=      1.0004670166177
afa= 0.0001000000    k=  6   U comp. Taylor test=      1.0004670166177
afa= 0.0001000000    k=  7   U comp. Taylor test=      1.0004670166177
afa= 0.0001000000    k=  8   U comp. Taylor test=      1.0004670166177
afa= 0.0001000000    k=  9   U comp. Taylor test=      1.0004670166177
afa= 0.0001000000    k= 10   U comp. Taylor test=      1.0004670166177
afa= 0.0001000000    k= 11   U comp. Taylor test=      1.0004670166177
afa= 0.0001000000    k= 12   U comp. Taylor test=      1.0004670166177
afa= 0.0001000000    k= 13   U comp. Taylor test=      1.0004670166177
afa= 0.0001000000    k= 14   U comp. Taylor test=      1.0004670166177
afa= 0.0001000000    k= 15   U comp. Taylor test=      1.0004670166177
afa= 0.0001000000    k= 16   U comp. Taylor test=      1.0004670166177
afa= 0.0001000000    k= 17   U comp. Taylor test=      1.0004670166177
afa= 0.0001000000    k= 18   U comp. Taylor test=      1.0004670166177
afa= 0.0001000000    k=  1   V comp. Taylor test=      0.9997552211761
afa= 0.0001000000    k=  2   V comp. Taylor test=      0.9997549384811
afa= 0.0001000000    k=  3   V comp. Taylor test=      0.9997549384811
afa= 0.0001000000    k=  4   V comp. Taylor test=      0.9997549384811
afa= 0.0001000000    k=  5   V comp. Taylor test=      0.9997549384811
afa= 0.0001000000    k=  6   V comp. Taylor test=      0.9997549384811
afa= 0.0001000000    k=  7   V comp. Taylor test=      0.9997549384811
afa= 0.0001000000    k=  8   V comp. Taylor test=      0.9997549384811
afa= 0.0001000000    k=  9   V comp. Taylor test=      0.9997549384811
afa= 0.0001000000    k= 10   V comp. Taylor test=      0.9997549384811
afa= 0.0001000000    k= 11   V comp. Taylor test=      0.9997549384811
afa= 0.0001000000    k= 12   V comp. Taylor test=      0.9997549384811
afa= 0.0001000000    k= 13   V comp. Taylor test=      0.9997549384811
afa= 0.0001000000    k= 14   V comp. Taylor test=      0.9997549384811
afa= 0.0001000000    k= 15   V comp. Taylor test=      0.9997549384811
afa= 0.0001000000    k= 16   V comp. Taylor test=      0.9997549384811
afa= 0.0001000000    k= 17   V comp. Taylor test=      0.9997549384811
afa= 0.0001000000    k= 18   V comp. Taylor test=      0.9997549384811
afa= 0.0001000000    k=  1   T comp. Taylor test=      0.9999996946707
afa= 0.0001000000    k=  2   T comp. Taylor test=      1.0000000455702
afa= 0.0001000000    k=  3   T comp. Taylor test=      1.0000000455702
```

```
 afa= 0.0001000000    k=  4   T comp. Taylor test=      1.0000000455702
 afa= 0.0001000000    k=  5   T comp. Taylor test=      1.0000000455702
 afa= 0.0001000000    k=  6   T comp. Taylor test=      1.0000000455702
 afa= 0.0001000000    k=  7   T comp. Taylor test=      1.0000000455702
 afa= 0.0001000000    k=  8   T comp. Taylor test=      1.0000000455702
 afa= 0.0001000000    k=  9   T comp. Taylor test=      1.0000000455702
 afa= 0.0001000000    k= 10   T comp. Taylor test=      1.0000000455702
 afa= 0.0001000000    k= 11   T comp. Taylor test=      1.0000000455702
 afa= 0.0001000000    k= 12   T comp. Taylor test=      1.0000000455702
 afa= 0.0001000000    k= 13   T comp. Taylor test=      1.0000000455702
 afa= 0.0001000000    k= 14   T comp. Taylor test=      1.0000000455702
 afa= 0.0001000000    k= 15   T comp. Taylor test=      1.0000000455702
 afa= 0.0001000000    k= 16   T comp. Taylor test=      1.0000000455702
 afa= 0.0001000000    k= 17   T comp. Taylor test=      1.0000000455702
 afa= 0.0001000000    k= 18   T comp. Taylor test=      1.0000000455702
 afa= 0.0001000000    k=  1   Q comp. Taylor test=      0.9999965949256
 afa= 0.0001000000    k=  2   Q comp. Taylor test=      0.9999965949256
 afa= 0.0001000000    k=  3   Q comp. Taylor test=      0.9999965949256
 afa= 0.0001000000    k=  4   Q comp. Taylor test=      0.9999965949256
 afa= 0.0001000000    k=  5   Q comp. Taylor test=      0.9999965949256
 afa= 0.0001000000    k=  6   Q comp. Taylor test=      0.9999965949256
 afa= 0.0001000000    k=  7   Q comp. Taylor test=      0.9999965949256
 afa= 0.0001000000    k=  8   Q comp. Taylor test=      0.9999965949256
 afa= 0.0001000000    k=  9   Q comp. Taylor test=      0.9999965949256
 afa= 0.0001000000    k= 10   Q comp. Taylor test=      0.9999965949256
 afa= 0.0001000000    k= 11   Q comp. Taylor test=      0.9999965949256
 afa= 0.0001000000    k= 12   Q comp. Taylor test=      0.9999965949256
 afa= 0.0001000000    k= 13   Q comp. Taylor test=      0.9999965949256
 afa= 0.0001000000    k= 14   Q comp. Taylor test=      0.9999965949256
 afa= 0.0001000000    k= 15   Q comp. Taylor test=      0.9999965949256
 afa= 0.0001000000    k= 16   Q comp. Taylor test=      0.9999965949256
 afa= 0.0001000000    k= 17   Q comp. Taylor test=      0.9999965949256
 afa= 0.0001000000    k= 18   Q comp. Taylor test=      0.9999965949256
 afa= 0.0001000000    Ps comp. Taylor test=      1.0000011009871
 afa= 0.0001000000    Taylor test=      0.9999999846721
 afa= 0.0000100000    k=  1   U comp. Taylor test=      1.0005494896200
 afa= 0.0000100000    k=  2   U comp. Taylor test=      1.0005391202092
 afa= 0.0000100000    k=  3   U comp. Taylor test=      1.0005391202092
 afa= 0.0000100000    k=  4   U comp. Taylor test=      1.0005391202092
 afa= 0.0000100000    k=  5   U comp. Taylor test=      1.0005391202092
 afa= 0.0000100000    k=  6   U comp. Taylor test=      1.0005391202092
 afa= 0.0000100000    k=  7   U comp. Taylor test=      1.0005391202092
 afa= 0.0000100000    k=  8   U comp. Taylor test=      1.0005391202092
 afa= 0.0000100000    k=  9   U comp. Taylor test=      1.0005391202092
 afa= 0.0000100000    k= 10   U comp. Taylor test=      1.0005391202092
 afa= 0.0000100000    k= 11   U comp. Taylor test=      1.0005391202092
 afa= 0.0000100000    k= 12   U comp. Taylor test=      1.0005391202092
 afa= 0.0000100000    k= 13   U comp. Taylor test=      1.0005391202092
 afa= 0.0000100000    k= 14   U comp. Taylor test=      1.0005391202092
 afa= 0.0000100000    k= 15   U comp. Taylor test=      1.0005391202092
 afa= 0.0000100000    k= 16   U comp. Taylor test=      1.0005391202092
 afa= 0.0000100000    k= 17   U comp. Taylor test=      1.0005391202092
 afa= 0.0000100000    k= 18   U comp. Taylor test=      0.9997580546348
 afa= 0.0000100000    k=  1   V comp. Taylor test=      0.9997685326328
 afa= 0.0000100000    k=  2   V comp. Taylor test=      0.9997685326328
 afa= 0.0000100000    k=  3   V comp. Taylor test=      0.9997685326328
```



```
 afa= 0.0000100000    k=  4   V comp. Taylor test=      0.9997685326328
 afa= 0.0000100000    k=  5   V comp. Taylor test=      0.9997685326328
 afa= 0.0000100000    k=  6   V comp. Taylor test=      0.9997685326328
 afa= 0.0000100000    k=  7   V comp. Taylor test=      0.9997685326328
 afa= 0.0000100000    k=  8   V comp. Taylor test=      0.9997685326328
 afa= 0.0000100000    k=  9   V comp. Taylor test=      0.9997685326328
 afa= 0.0000100000    k= 10   V comp. Taylor test=      0.9997685326328
 afa= 0.0000100000    k= 11   V comp. Taylor test=      0.9997685326328
 afa= 0.0000100000    k= 12   V comp. Taylor test=      0.9997685326328
 afa= 0.0000100000    k= 13   V comp. Taylor test=      0.9997685326328
 afa= 0.0000100000    k= 14   V comp. Taylor test=      0.9997685326328
 afa= 0.0000100000    k= 15   V comp. Taylor test=      0.9997685326328
 afa= 0.0000100000    k= 16   V comp. Taylor test=      0.9997685326328
 afa= 0.0000100000    k= 17   V comp. Taylor test=      0.9997685326328
 afa= 0.0000100000    k= 18   V comp. Taylor test=      0.9997685326328
 afa= 0.0000100000    k=  1   T comp. Taylor test=      0.9999994740654
 afa= 0.0000100000    k=  2   T comp. Taylor test=      0.9999999003721
 afa= 0.0000100000    k=  3   T comp. Taylor test=      0.9999999003721
 afa= 0.0000100000    k=  4   T comp. Taylor test=      0.9999999003721
 afa= 0.0000100000    k=  5   T comp. Taylor test=      0.9999999003721
 afa= 0.0000100000    k=  6   T comp. Taylor test=      0.9999999003721
 afa= 0.0000100000    k=  7   T comp. Taylor test=      0.9999999003721
 afa= 0.0000100000    k=  8   T comp. Taylor test=      0.9999999003721
 afa= 0.0000100000    k=  9   T comp. Taylor test=      0.9999999003721
 afa= 0.0000100000    k= 10   T comp. Taylor test=      0.9999999003721
 afa= 0.0000100000    k= 11   T comp. Taylor test=      0.9999999003721
 afa= 0.0000100000    k= 12   T comp. Taylor test=      0.9999999003721
 afa= 0.0000100000    k= 13   T comp. Taylor test=      0.9999999003721
 afa= 0.0000100000    k= 14   T comp. Taylor test=      0.9999999003721
 afa= 0.0000100000    k= 15   T comp. Taylor test=      0.9999999003721
 afa= 0.0000100000    k= 16   T comp. Taylor test=      0.9999999003721
 afa= 0.0000100000    k= 17   T comp. Taylor test=      0.9999999003721
 afa= 0.0000100000    k= 18   T comp. Taylor test=      0.9999999003721
 afa= 0.0000100000    k=  1   Q comp. Taylor test=      0.9999944386767
 afa= 0.0000100000    k=  2   Q comp. Taylor test=      0.9999944386767
 afa= 0.0000100000    k=  3   Q comp. Taylor test=      0.9999944386767
 afa= 0.0000100000    k=  4   Q comp. Taylor test=      0.9999944386767
 afa= 0.0000100000    k=  5   Q comp. Taylor test=      0.9999944386767
 afa= 0.0000100000    k=  6   Q comp. Taylor test=      0.9999944386767
 afa= 0.0000100000    k=  7   Q comp. Taylor test=      0.9999944386767
 afa= 0.0000100000    k=  8   Q comp. Taylor test=      0.9999944386767
 afa= 0.0000100000    k=  9   Q comp. Taylor test=      0.9999944386767
 afa= 0.0000100000    k= 10   Q comp. Taylor test=      0.9999944386767
 afa= 0.0000100000    k= 11   Q comp. Taylor test=      0.9999944386767
 afa= 0.0000100000    k= 12   Q comp. Taylor test=      0.9999944386767
 afa= 0.0000100000    k= 13   Q comp. Taylor test=      0.9999944386767
 afa= 0.0000100000    k= 14   Q comp. Taylor test=      0.9999944386767
 afa= 0.0000100000    k= 15   Q comp. Taylor test=      0.9999944386767
 afa= 0.0000100000    k= 16   Q comp. Taylor test=      0.9999944386767
 afa= 0.0000100000    k= 17   Q comp. Taylor test=      0.9999944386767
 afa= 0.0000100000    k= 18   Q comp. Taylor test=      0.9999944386767
 afa= 0.0000100000    Ps comp. Taylor test=      1.0000011270335
 afa= 0.0000010000    Taylor test=      1.0000000826829
 afa= 0.0000010000    k=  1   U comp. Taylor test=      1.0010518966339
 afa= 0.0000010000    k=  2   U comp. Taylor test=      1.0006858637753
 afa= 0.0000010000    k=  3   U comp. Taylor test=      1.0006858637753
```

```
 afa= 0.0000010000    k=  4   U comp. Taylor test=      1.0006858637753
 afa= 0.0000010000    k=  5   U comp. Taylor test=      1.0006858637753
 afa= 0.0000010000    k=  6   U comp. Taylor test=      1.0006858637753
 afa= 0.0000010000    k=  7   U comp. Taylor test=      1.0006858637753
 afa= 0.0000010000    k=  8   U comp. Taylor test=      1.0006858637753
 afa= 0.0000010000    k=  9   U comp. Taylor test=      1.0006858637753
 afa= 0.0000010000    k= 10   U comp. Taylor test=      1.0006858637753
 afa= 0.0000010000    k= 11   U comp. Taylor test=      1.0006858637753
 afa= 0.0000010000    k= 12   U comp. Taylor test=      1.0006858637753
 afa= 0.0000010000    k= 13   U comp. Taylor test=      1.0006858637753
 afa= 0.0000010000    k= 14   U comp. Taylor test=      1.0006858637753
 afa= 0.0000010000    k= 15   U comp. Taylor test=      1.0006858637753
 afa= 0.0000010000    k= 16   U comp. Taylor test=      1.0006858637753
 afa= 0.0000010000    k= 17   U comp. Taylor test=      1.0006858637753
 afa= 0.0000010000    k= 18   U comp. Taylor test=      1.0006858637753
 afa= 0.0000010000    k=  1   V comp. Taylor test=      0.9995853027674
 afa= 0.0000010000    k=  2   V comp. Taylor test=      0.9997584170596
 afa= 0.0000010000    k=  3   V comp. Taylor test=      0.9997584170596
 afa= 0.0000010000    k=  4   V comp. Taylor test=      0.9997584170596
 afa= 0.0000010000    k=  5   V comp. Taylor test=      0.9997584170596
 afa= 0.0000010000    k=  6   V comp. Taylor test=      0.9997584170596
 afa= 0.0000010000    k=  7   V comp. Taylor test=      0.9997584170596
 afa= 0.0000010000    k=  8   V comp. Taylor test=      0.9997584170596
 afa= 0.0000010000    k=  9   V comp. Taylor test=      0.9997584170596
 afa= 0.0000010000    k= 10   V comp. Taylor test=      0.9997584170596
 afa= 0.0000010000    k= 11   V comp. Taylor test=      0.9997584170596
 afa= 0.0000010000    k= 12   V comp. Taylor test=      0.9997584170596
 afa= 0.0000010000    k= 13   V comp. Taylor test=      0.9997584170596
 afa= 0.0000010000    k= 14   V comp. Taylor test=      0.9997584170596
 afa= 0.0000010000    k= 15   V comp. Taylor test=      0.9997584170596
 afa= 0.0000010000    k= 16   V comp. Taylor test=      0.9997584170596
 afa= 0.0000010000    k= 17   V comp. Taylor test=      0.9997584170596
 afa= 0.0000010000    k= 18   V comp. Taylor test=      0.9997584170596
 afa= 0.0000010000    k=  1   T comp. Taylor test=      0.9999987549011
 afa= 0.0000010000    k=  2   T comp. Taylor test=      0.9999999081724
 afa= 0.0000010000    k=  3   T comp. Taylor test=      0.9999999081724
 afa= 0.0000010000    k=  4   T comp. Taylor test=      0.9999999081724
 afa= 0.0000010000    k=  5   T comp. Taylor test=      0.9999999081724
 afa= 0.0000010000    k=  6   T comp. Taylor test=      0.9999999081724
 afa= 0.0000010000    k=  7   T comp. Taylor test=      0.9999999081724
 afa= 0.0000010000    k=  8   T comp. Taylor test=      0.9999999081724
 afa= 0.0000010000    k=  9   T comp. Taylor test=      0.9999999081724
 afa= 0.0000010000    k= 10   T comp. Taylor test=      0.9999999081724
 afa= 0.0000010000    k= 11   T comp. Taylor test=      0.9999999081724
 afa= 0.0000010000    k= 12   T comp. Taylor test=      0.9999999081724
 afa= 0.0000010000    k= 13   T comp. Taylor test=      0.9999999081724
 afa= 0.0000010000    k= 14   T comp. Taylor test=      0.9999999081724
 afa= 0.0000010000    k= 15   T comp. Taylor test=      0.9999999081724
 afa= 0.0000010000    k= 16   T comp. Taylor test=      0.9999999081724
 afa= 0.0000010000    k= 17   T comp. Taylor test=      0.9999999081724
 afa= 0.0000010000    k= 18   T comp. Taylor test=      0.9999999081724
 afa= 0.0000010000    k=  1   Q comp. Taylor test=      0.9999821219629
 afa= 0.0000010000    k=  2   Q comp. Taylor test=      0.9999821219629
 afa= 0.0000010000    k=  3   Q comp. Taylor test=      0.9999821219629
 afa= 0.0000010000    k=  4   Q comp. Taylor test=      0.9999821219629
 afa= 0.0000010000    k=  5   Q comp. Taylor test=      0.9999821219629
```



```
 afa= 0.0000010000    k=  6   Q comp. Taylor test=      0.9999821219629
 afa= 0.0000010000    k=  7   Q comp. Taylor test=      0.9999821219629
 afa= 0.0000010000    k=  8   Q comp. Taylor test=      0.9999821219629
 afa= 0.0000010000    k=  9   Q comp. Taylor test=      0.9999821219629
 afa= 0.0000010000    k= 10   Q comp. Taylor test=      0.9999821219629
 afa= 0.0000010000    k= 11   Q comp. Taylor test=      0.9999821219629
 afa= 0.0000010000    k= 12   Q comp. Taylor test=      0.9999821219629
 afa= 0.0000010000    k= 13   Q comp. Taylor test=      0.9999821219629
 afa= 0.0000010000    k= 14   Q comp. Taylor test=      0.9999821219629
 afa= 0.0000010000    k= 15   Q comp. Taylor test=      0.9999821219629
 afa= 0.0000010000    k= 16   Q comp. Taylor test=      0.9999821219629
 afa= 0.0000010000    k= 17   Q comp. Taylor test=      0.9999821219629
 afa= 0.0000010000    k= 18   Q comp. Taylor test=      0.9999821219629
 afa= 0.0000010000    Ps comp. Taylor test=      1.0000024474998
 afa= 0.0000001000    Taylor test=      0.9999773922282
 afa= 0.0000001000    k=  1   U comp. Taylor test=      1.0036854572158
 afa= 0.0000001000    k=  2   U comp. Taylor test=      1.0029971074081
 afa= 0.0000001000    k=  3   U comp. Taylor test=      1.0029971074081
 afa= 0.0000001000    k=  4   U comp. Taylor test=      1.0029971074081
 afa= 0.0000001000    k=  5   U comp. Taylor test=      1.0029971074081
 afa= 0.0000001000    k=  6   U comp. Taylor test=      1.0029971074081
 afa= 0.0000001000    k=  7   U comp. Taylor test=      1.0029971074081
 afa= 0.0000001000    k=  8   U comp. Taylor test=      1.0029971074081
 afa= 0.0000001000    k=  9   U comp. Taylor test=      1.0029971074081
 afa= 0.0000001000    k= 10   U comp. Taylor test=      1.0029971074081
 afa= 0.0000001000    k= 11   U comp. Taylor test=      1.0029971074081
 afa= 0.0000001000    k= 12   U comp. Taylor test=      1.0029971074081
 afa= 0.0000001000    k= 13   U comp. Taylor test=      1.0029971074081
 afa= 0.0000001000    k= 14   U comp. Taylor test=      1.0029971074081
 afa= 0.0000001000    k= 15   U comp. Taylor test=      1.0029971074081
 afa= 0.0000001000    k= 16   U comp. Taylor test=      1.0029971074081
 afa= 0.0000001000    k= 17   U comp. Taylor test=      1.0029971074081
 afa= 0.0000001000    k= 18   U comp. Taylor test=      1.0029971074081
 afa= 0.0000001000    k=  1   V comp. Taylor test=      0.9976630793301
 afa= 0.0000001000    k=  2   V comp. Taylor test=      1.0013300133441
 afa= 0.0000001000    k=  3   V comp. Taylor test=      1.0013300133441
 afa= 0.0000001000    k=  4   V comp. Taylor test=      1.0013300133441
 afa= 0.0000001000    k=  5   V comp. Taylor test=      1.0013300133441
 afa= 0.0000001000    k=  6   V comp. Taylor test=      1.0013300133441
 afa= 0.0000001000    k=  7   V comp. Taylor test=      1.0013300133441
 afa= 0.0000001000    k=  8   V comp. Taylor test=      1.0013300133441
 afa= 0.0000001000    k=  9   V comp. Taylor test=      1.0013300133441
 afa= 0.0000001000    k= 10   V comp. Taylor test=      1.0013300133441
 afa= 0.0000001000    k= 11   V comp. Taylor test=      1.0013300133441
 afa= 0.0000001000    k= 12   V comp. Taylor test=      1.0013300133441
 afa= 0.0000001000    k= 13   V comp. Taylor test=      1.0013300133441
 afa= 0.0000001000    k= 14   V comp. Taylor test=      1.0013300133441
 afa= 0.0000001000    k= 15   V comp. Taylor test=      1.0013300133441
 afa= 0.0000001000    k= 16   V comp. Taylor test=      1.0013300133441
 afa= 0.0000001000    k= 17   V comp. Taylor test=      1.0013300133441
 afa= 0.0000001000    k= 18   V comp. Taylor test=      1.0013300133441
 afa= 0.0000001000    k=  1   T comp. Taylor test=      0.9999679360965
 afa= 0.0000001000    k=  2   T comp. Taylor test=      0.9999752403410
 afa= 0.0000001000    k=  3   T comp. Taylor test=      0.9999752403410
 afa= 0.0000001000    k=  4   T comp. Taylor test=      0.9999752403410
 afa= 0.0000001000    k=  5   T comp. Taylor test=      0.9999752403410
```

```
 afa= 0.0000001000    k=  6   T comp. Taylor test=      0.9999752403410
 afa= 0.0000001000    k=  7   T comp. Taylor test=      0.9999752403410
 afa= 0.0000001000    k=  8   T comp. Taylor test=      0.9999752403410
 afa= 0.0000001000    k=  9   T comp. Taylor test=      0.9999752403410
 afa= 0.0000001000    k= 10   T comp. Taylor test=      0.9999752403410
 afa= 0.0000001000    k= 11   T comp. Taylor test=      0.9999752403410
 afa= 0.0000001000    k= 12   T comp. Taylor test=      0.9999752403410
 afa= 0.0000001000    k= 13   T comp. Taylor test=      0.9999752403410
 afa= 0.0000001000    k= 14   T comp. Taylor test=      0.9999752403410
 afa= 0.0000001000    k= 15   T comp. Taylor test=      0.9999752403410
 afa= 0.0000001000    k= 16   T comp. Taylor test=      0.9999752403410
 afa= 0.0000001000    k= 17   T comp. Taylor test=      0.9999752403410
 afa= 0.0000001000    k= 18   T comp. Taylor test=      0.9997907832299
 afa= 0.0000001000    k=  1   Q comp. Taylor test=      0.9997907832299
 afa= 0.0000001000    k=  2   Q comp. Taylor test=      0.9997907832299
 afa= 0.0000001000    k=  3   Q comp. Taylor test=      0.9997907832299
 afa= 0.0000001000    k=  4   Q comp. Taylor test=      0.9997907832299
 afa= 0.0000001000    k=  5   Q comp. Taylor test=      0.9997907832299
 afa= 0.0000001000    k=  6   Q comp. Taylor test=      0.9997907832299
 afa= 0.0000001000    k=  7   Q comp. Taylor test=      0.9997907832299
 afa= 0.0000001000    k=  8   Q comp. Taylor test=      0.9997907832299
 afa= 0.0000001000    k=  9   Q comp. Taylor test=      0.9997907832299
 afa= 0.0000001000    k= 10   Q comp. Taylor test=      0.9997907832299
 afa= 0.0000001000    k= 11   Q comp. Taylor test=      0.9997907832299
 afa= 0.0000001000    k= 12   Q comp. Taylor test=      0.9997907832299
 afa= 0.0000001000    k= 13   Q comp. Taylor test=      0.9997907832299
 afa= 0.0000001000    k= 14   Q comp. Taylor test=      0.9997907832299
 afa= 0.0000001000    k= 15   Q comp. Taylor test=      0.9997907832299
 afa= 0.0000001000    k= 16   Q comp. Taylor test=      0.9997907832299
 afa= 0.0000001000    k= 17   Q comp. Taylor test=      0.9997907832299
 afa= 0.0000001000    k= 18   Q comp. Taylor test=      0.9997907832299
 afa= 0.0000001000    Ps comp. Taylor test=      1.0000065523513
                      Taylor test=      1.0000762269526
 afa= 0.0000000100    k=  1   U comp. Taylor test=      1.3447566674003
 afa= 0.0000000100    k=  2   U comp. Taylor test=      1.3103065863120
 afa= 0.0000000100    k=  3   U comp. Taylor test=      1.3103065863120
 afa= 0.0000000100    k=  4   U comp. Taylor test=      1.3103065863120
 afa= 0.0000000100    k=  5   U comp. Taylor test=      1.3103065863120
 afa= 0.0000000100    k=  6   U comp. Taylor test=      1.3103065863120
 afa= 0.0000000100    k=  7   U comp. Taylor test=      1.3103065863120
 afa= 0.0000000100    k=  8   U comp. Taylor test=      1.3103065863120
 afa= 0.0000000100    k=  9   U comp. Taylor test=      1.3103065863120
 afa= 0.0000000100    k= 10   U comp. Taylor test=      1.3103065863120
 afa= 0.0000000100    k= 11   U comp. Taylor test=      1.3103065863120
 afa= 0.0000000100    k= 12   U comp. Taylor test=      1.3103065863120
 afa= 0.0000000100    k= 13   U comp. Taylor test=      1.3103065863120
 afa= 0.0000000100    k= 14   U comp. Taylor test=      1.3103065863120
 afa= 0.0000000100    k= 15   U comp. Taylor test=      1.3103065863120
 afa= 0.0000000100    k= 16   U comp. Taylor test=      1.3103065863120
 afa= 0.0000000100    k= 17   U comp. Taylor test=      1.3103065863120
 afa= 0.0000000100    k= 18   U comp. Taylor test=      1.1371569547529
 afa= 0.0000000100    k=  1   V comp. Taylor test=      1.1604740640971
 afa= 0.0000000100    k=  2   V comp. Taylor test=      1.1604740640971
 afa= 0.0000000100    k=  3   V comp. Taylor test=      1.1604740640971
 afa= 0.0000000100    k=  4   V comp. Taylor test=      1.1604740640971
 afa= 0.0000000100    k=  5   V comp. Taylor test=      1.1604740640971
```



```
 afa= 0.0000000100    k=  6   V comp. Taylor test=      1.1604740640971
 afa= 0.0000000100    k=  7   V comp. Taylor test=      1.1604740640971
 afa= 0.0000000100    k=  8   V comp. Taylor test=      1.1604740640971
 afa= 0.0000000100    k=  9   V comp. Taylor test=      1.1604740640971
 afa= 0.0000000100    k= 10   V comp. Taylor test=      1.1604740640971
 afa= 0.0000000100    k= 11   V comp. Taylor test=      1.1604740640971
 afa= 0.0000000100    k= 12   V comp. Taylor test=      1.1604740640971
 afa= 0.0000000100    k= 13   V comp. Taylor test=      1.1604740640971
 afa= 0.0000000100    k= 14   V comp. Taylor test=      1.1604740640971
 afa= 0.0000000100    k= 15   V comp. Taylor test=      1.1604740640971
 afa= 0.0000000100    k= 16   V comp. Taylor test=      1.1604740640971
 afa= 0.0000000100    k= 17   V comp. Taylor test=      1.1604740640971
 afa= 0.0000000100    k= 18   V comp. Taylor test=      1.0000051718550
 afa= 0.0000000100    k=  1   T comp. Taylor test=      1.0000525699912
 afa= 0.0000000100    k=  2   T comp. Taylor test=      1.0000525699912
 afa= 0.0000000100    k=  3   T comp. Taylor test=      1.0000525699912
 afa= 0.0000000100    k=  4   T comp. Taylor test=      1.0000525699912
 afa= 0.0000000100    k=  5   T comp. Taylor test=      1.0000525699912
 afa= 0.0000000100    k=  6   T comp. Taylor test=      1.0000525699912
 afa= 0.0000000100    k=  7   T comp. Taylor test=      1.0000525699912
 afa= 0.0000000100    k=  8   T comp. Taylor test=      1.0000525699912
 afa= 0.0000000100    k=  9   T comp. Taylor test=      1.0000525699912
 afa= 0.0000000100    k= 10   T comp. Taylor test=      1.0000525699912
 afa= 0.0000000100    k= 11   T comp. Taylor test=      1.0000525699912
 afa= 0.0000000100    k= 12   T comp. Taylor test=      1.0000525699912
 afa= 0.0000000100    k= 13   T comp. Taylor test=      1.0000525699912
 afa= 0.0000000100    k= 14   T comp. Taylor test=      1.0000525699912
 afa= 0.0000000100    k= 15   T comp. Taylor test=      1.0000525699912
 afa= 0.0000000100    k= 16   T comp. Taylor test=      1.0000525699912
 afa= 0.0000000100    k= 17   T comp. Taylor test=      1.0000525699912
 afa= 0.0000000100    k= 18   T comp. Taylor test=      1.0000525699912
 afa= 0.0000000100    k=  1   Q comp. Taylor test=      0.9990829358147
 afa= 0.0000000100    k=  2   Q comp. Taylor test=      0.9990829358147
 afa= 0.0000000100    k=  3   Q comp. Taylor test=      0.9990829358147
 afa= 0.0000000100    k=  4   Q comp. Taylor test=      0.9990829358147
 afa= 0.0000000100    k=  5   Q comp. Taylor test=      0.9990829358147
 afa= 0.0000000100    k=  6   Q comp. Taylor test=      0.9990829358147
 afa= 0.0000000100    k=  7   Q comp. Taylor test=      0.9990829358147
 afa= 0.0000000100    k=  8   Q comp. Taylor test=      0.9990829358147
 afa= 0.0000000100    k=  9   Q comp. Taylor test=      0.9990829358147
 afa= 0.0000000100    k= 10   Q comp. Taylor test=      0.9990829358147
 afa= 0.0000000100    k= 11   Q comp. Taylor test=      0.9990829358147
 afa= 0.0000000100    k= 12   Q comp. Taylor test=      0.9990829358147
 afa= 0.0000000100    k= 13   Q comp. Taylor test=      0.9990829358147
 afa= 0.0000000100    k= 14   Q comp. Taylor test=      0.9990829358147
 afa= 0.0000000100    k= 15   Q comp. Taylor test=      0.9990829358147
 afa= 0.0000000100    k= 16   Q comp. Taylor test=      0.9990829358147
 afa= 0.0000000100    k= 17   Q comp. Taylor test=      0.9990829358147
 afa= 0.0000000100    k= 18   Q comp. Taylor test=      0.9990829358147
 afa= 0.0000000100    Ps comp. Taylor test=      1.0003967512070
                      Taylor test=      0.9993970130244
 afa= 0.0000000010    k=  1   U comp. Taylor test=      7.8322659778035
 afa= 0.0000000010    k=  2   U comp. Taylor test=      8.1819018619861
 afa= 0.0000000010    k=  3   U comp. Taylor test=      8.1819018619861
 afa= 0.0000000010    k=  4   U comp. Taylor test=      8.1819018619861
 afa= 0.0000000010    k=  5   U comp. Taylor test=      8.1819018619861
```

```
 afa= 0.0000000010    k=  6   U comp. Taylor test=      8.1819018619861
 afa= 0.0000000010    k=  7   U comp. Taylor test=      8.1819018619861
 afa= 0.0000000010    k=  8   U comp. Taylor test=      8.1819018619861
 afa= 0.0000000010    k=  9   U comp. Taylor test=      8.1819018619861
 afa= 0.0000000010    k= 10   U comp. Taylor test=      8.1819018619861
 afa= 0.0000000010    k= 11   U comp. Taylor test=      8.1819018619861
 afa= 0.0000000010    k= 12   U comp. Taylor test=      8.1819018619861
 afa= 0.0000000010    k= 13   U comp. Taylor test=      8.1819018619861
 afa= 0.0000000010    k= 14   U comp. Taylor test=      8.1819018619861
 afa= 0.0000000010    k= 15   U comp. Taylor test=      8.1819018619861
 afa= 0.0000000010    k= 16   U comp. Taylor test=      8.1819018619861
 afa= 0.0000000010    k= 17   U comp. Taylor test=      8.1819018619861
 afa= 0.0000000010    k= 18   U comp. Taylor test=      8.1819018619861
 afa= 0.0000000010    k=  1   V comp. Taylor test=      6.3077206161025
 afa= 0.0000000010    k=  2   V comp. Taylor test=      6.9840147008458
 afa= 0.0000000010    k=  3   V comp. Taylor test=      6.9840147008458
 afa= 0.0000000010    k=  4   V comp. Taylor test=      6.9840147008458
 afa= 0.0000000010    k=  5   V comp. Taylor test=      6.9840147008458
 afa= 0.0000000010    k=  6   V comp. Taylor test=      6.9840147008458
 afa= 0.0000000010    k=  7   V comp. Taylor test=      6.9840147008458
 afa= 0.0000000010    k=  8   V comp. Taylor test=      6.9840147008458
 afa= 0.0000000010    k=  9   V comp. Taylor test=      6.9840147008458
 afa= 0.0000000010    k= 10   V comp. Taylor test=      6.9840147008458
 afa= 0.0000000010    k= 11   V comp. Taylor test=      6.9840147008458
 afa= 0.0000000010    k= 12   V comp. Taylor test=      6.9840147008458
 afa= 0.0000000010    k= 13   V comp. Taylor test=      6.9840147008458
 afa= 0.0000000010    k= 14   V comp. Taylor test=      6.9840147008458
 afa= 0.0000000010    k= 15   V comp. Taylor test=      6.9840147008458
 afa= 0.0000000010    k= 16   V comp. Taylor test=      6.9840147008458
 afa= 0.0000000010    k= 17   V comp. Taylor test=      6.9840147008458
 afa= 0.0000000010    k= 18   V comp. Taylor test=      6.9840147008458
 afa= 0.0000000010    k=  1   T comp. Taylor test=      0.9986944953887
 afa= 0.0000000010    k=  2   T comp. Taylor test=      0.9991971470146
 afa= 0.0000000010    k=  3   T comp. Taylor test=      0.9991971470146
 afa= 0.0000000010    k=  4   T comp. Taylor test=      0.9991971470146
 afa= 0.0000000010    k=  5   T comp. Taylor test=      0.9991971470146
 afa= 0.0000000010    k=  6   T comp. Taylor test=      0.9991971470146
 afa= 0.0000000010    k=  7   T comp. Taylor test=      0.9991971470146
 afa= 0.0000000010    k=  8   T comp. Taylor test=      0.9991971470146
 afa= 0.0000000010    k=  9   T comp. Taylor test=      0.9991971470146
 afa= 0.0000000010    k= 10   T comp. Taylor test=      0.9991971470146
 afa= 0.0000000010    k= 11   T comp. Taylor test=      0.9991971470146
 afa= 0.0000000010    k= 12   T comp. Taylor test=      0.9991971470146
 afa= 0.0000000010    k= 13   T comp. Taylor test=      0.9991971470146
 afa= 0.0000000010    k= 14   T comp. Taylor test=      0.9991971470146
 afa= 0.0000000010    k= 15   T comp. Taylor test=      0.9991971470146
 afa= 0.0000000010    k= 16   T comp. Taylor test=      0.9991971470146
 afa= 0.0000000010    k= 17   T comp. Taylor test=      0.9991971470146
 afa= 0.0000000010    k= 18   T comp. Taylor test=      0.9991971470146
 afa= 0.0000000010    k=  1   Q comp. Taylor test=      0.9826753974247
 afa= 0.0000000010    k=  2   Q comp. Taylor test=      0.9826753974247
 afa= 0.0000000010    k=  3   Q comp. Taylor test=      0.9826753974247
 afa= 0.0000000010    k=  4   Q comp. Taylor test=      0.9826753974247
 afa= 0.0000000010    k=  5   Q comp. Taylor test=      0.9826753974247
 afa= 0.0000000010    k=  6   Q comp. Taylor test=      0.9826753974247
 afa= 0.0000000010    k=  7   Q comp. Taylor test=      0.9826753974247
```



```
 afa= 0.0000000010    k=  8   Q comp. Taylor test=      0.9826753974247
 afa= 0.0000000010    k=  9   Q comp. Taylor test=      0.9826753974247
 afa= 0.0000000010    k= 10   Q comp. Taylor test=      0.9826753974247
 afa= 0.0000000010    k= 11   Q comp. Taylor test=      0.9826753974247
 afa= 0.0000000010    k= 12   Q comp. Taylor test=      0.9826753974247
 afa= 0.0000000010    k= 13   Q comp. Taylor test=      0.9826753974247
 afa= 0.0000000010    k= 14   Q comp. Taylor test=      0.9826753974247
 afa= 0.0000000010    k= 15   Q comp. Taylor test=      0.9826753974247
 afa= 0.0000000010    k= 16   Q comp. Taylor test=      0.9826753974247
 afa= 0.0000000010    k= 17   Q comp. Taylor test=      0.9826753974247
 afa= 0.0000000010    k= 18   Q comp. Taylor test=      0.9826753974247
 afa= 0.0000000010    Ps comp. Taylor test=      1.0021015003357
```

表 2

User gfs11@as06 Apr 26 04:06 1997 adtime.f Page 1

```
      program adtime
      real xold,xnow,dt,a,afa
      afa=0.02
      dt=2
      a=0.3
      xnow=3
      axx=xnow
      aa=a
      call ttt(a,dt,afa)
      w=a*a
      print *,'a= ',a
      print *,'w= ',w
      call adttt(a,dt,afa)
      v=a*aa
      print *,'----------------------------'
      print *,'w,v= ',w,v
      stop
      end
c
c
      subroutine ttt(a,dt,afa)
      real xold,a,xnow,dt,afa,xten
      logical forward
      data forward/.true./
      xnow=5.*a
      xold=xnow
      do i=1,5
      xten=a
      if (forward) then
      xnow=xold+dt*xten
      forward=.false.
      else
      xten=xold+2*dt*xten
      xold=xnow+afa*(xold-2*xnow+xten)
      xnow=xten
      endif
      a=4.*xnow
      enddo
      return
      end
c
c
      subroutine adttt(a,dt,afa)
      real xold,a,dt,afa,xnow,xten
      print *,'afa= ',afa
c
      xold=0.
      xnow=0.
      xten=0.
c
      do i=5,1,-1
      xnow=4.*a+xnow
      if (i.ne.1) then
      xten=xnow
      xnow=(1-2.*afa)*xold
      xten=afa*xold+xten
```

User gfs11@as06 Apr 26 04:06 1997 adtime.f Page 2

```
      xold=afa*xold
      xold=xten+xold
      xten=2*dt*xten
      print *,'1st xten= ',xten
c
      else
      xold=xnow+xold
      xten=dt*xnow
      print *,'2nd xten= ',xten
      xnow=0.
      endif
c
      a=xten
      enddo
      xnow=xold+xnow
      a=5.*xnow+a
      print *,'a,xnow= ',a,xnow
      return
      end
```

表三

全球動力波譜模式所用的副程式名

cons	eigen	gausl3	gfcst	gridnl	Mintgrt	invmtx	lgndr
Mmatrix	mtxmlp	mtxprt	Mpmatrx	Mprexp	rstran	Msfcdrg	Msiimpl
sortml	trandv	tranrs	transr	tranuv	trangra	uzmean	Mvstruc

正切線性模式所用的副程式名

cons	eigen	gausl3	Lgridnl	Lintgrt	invmtx	lgndr	Lmatrix
mtxmlp	mtxprt	Lpmatrx	Lprexp	rstran	Lsfcdrg	Lsiimpl	sortml
trandv	tranrs	transr	tranuv	trangra	uzmean	Lvstruc	

伴隨模式所用的副程式名

cons	eigen	gausl3	adLgridnl	adLintgrt	invmtx
lgndr	adLmatrix	mtxmlp	mtxprt	adLpmatrx	adLprexp
adrstran	adLsfcdrg	adLsiimpl	sortml	adtrandv	adtranrs
adtransr	adtranuv	adtrangra	uzmean	adLvstruc	

附錄


```
      program gfcstx_tlm
c
c Global ForeCaST Dry(x) Tangent Linear Model
c
      include '../include/param.h'
      include '../include/const.h'
      include '../include/gridtlm.h'
      include '../include/gridtlm9.h'
      include '../include/spec.h'
      include '../include/spec9.h'
c
      character*1 bas_tmpf_need
c
      dimension ut(nx,lev,my),vt(nx,lev,my),tt(nx,lev,my)
     >,         pt(nx,my),qt(nx,lev,my)
      dimension ut9(nx,lev,my),vt9(nx,lev,my),tt9(nx,lev,my)
     >,         pt9(nx,my),qt9(nx,lev,my)
      dimension ut99(nx,lev,my),vt99(nx,lev,my),tt99(nx,lev,my)
     >,         pt99(nx,my),qt99(nx,lev,my)
      dimension utdiff(nx,lev,my),vtdiff(nx,lev,my),ttdiff(nx,lev,my)
     >,         ptdiff(nx,my),qtdiff(nx,lev,my)
      dimension xx(nx*lev*my*4+nx*my),xx9(nx*lev*my*4+nx*my)
     >,         xx99(nx*lev*my*4+nx*my)
c
c
c  logical io units:
c
c  1:  input namelist file = 'namlsts'
c  3:  spectrum coeff output file='bspt??'
c 12:  filist
c
c  get model constants
c
      call cons
c
c  read in initial data and prepare for initialization/forecast
c
c  set up Basic state
c
      call grossby(nx,my,lev,sinl,tmean9,ut9,vt9,tt9,qt9,pt9
     *,            sig,ptop,pk9,pk29,plt9,tref9,capa,cosl)
c
c  set up perturb field
c
      call perturb(nx,my,lev,sinl,tmean9,tmean
     *,            pt9,pk9,pk29,tt9
     *,            ut,vt,tt,qt,pt,sig,ptop
     *,            pk,pk2,plt,tref,capa,cosl)
c
      print *,'finished setting initial field'
c
c  time integration
c  INPUT: ut,vt,tt,pt,qt
c                                  OUTPUT: ut,vt,tt,pt,qt
c
      bas_tmpf_need='y'
```



```
      call Mintgrt(ut9,vt9,tt9,pt9,qt9,bas_tmpf_need)
      call Lintgrt(ut,vt,tt,pt,qt)
      call trnv2x(ut9,vt9,tt9,pt9,qt9,nx,lev,my,xx9)
      call trnv2x(ut,vt,tt,pt,qt,nx,lev,my,xx)
c
      read(53,rec=1)ut9,vt9
      read(54,rec=1)tt9,qt9,pt9
      read(51,rec=1)ut,vt
      read(52,rec=1)tt,qt,pt
      bas_tmpf_need='n'
c
      afa=10
      do ia=10,1,-1
      afa=afa*0.1
c
      do j=1,my
      do i=1,nx
      do k=1,lev
      ut99(i,k,j)=ut9(i,k,j)+afa*ut(i,k,j)
      vt99(i,k,j)=vt9(i,k,j)+afa*vt(i,k,j)
      tt99(i,k,j)=tt9(i,k,j)+afa*tt(i,k,j)
      qt99(i,k,j)=qt9(i,k,j)+afa*qt(i,k,j)
      enddo
      pt99(i,j)=pt9(i,j)+afa*pt(i,j)
      enddo
      enddo
c
      call Mintgrt(ut99,vt99,tt99,pt99,qt99,bas_tmpf_need)
      call trnv2x(ut99,vt99,tt99,pt99,qt99,nx,lev,my,xx99)
c
      call Taylor_test(xx99,xx9,xx,nx,lev,my,afa)
      call Taylor_test_u(xx99,xx9,xx,nx,lev,my,afa)
      call Taylor_test_v(xx99,xx9,xx,nx,lev,my,afa)
      call Taylor_test_t(xx99,xx9,xx,nx,lev,my,afa)
      call Taylor_test_q(xx99,xx9,xx,nx,lev,my,afa)
      call Taylor_test_ps(xx99,xx9,xx,nx,lev,my,afa)
c
      if (ia.eq.10) then
      itaue=taue*3600./dt+0.001
      nrec_f=itaue+1.001
      print *,'nrec_f= ',nrec_f
      read(53,rec=nrec_f)ut9,vt9
      read(54,rec=nrec_f)tt9,qt9,pt9
      do j=1,my
      do i=1,nx
      do k=1,lev
      utdiff(i,k,j)=ut99(i,k,j)-ut9(i,k,j)
      vtdiff(i,k,j)=vt99(i,k,j)-vt9(i,k,j)
      ttdiff(i,k,j)=tt99(i,k,j)-tt9(i,k,j)
      qtdiff(i,k,j)=qt99(i,k,j)-qt9(i,k,j)
      enddo
      ptdiff(i,j)=pt99(i,j)-pt9(i,j)
      enddo
      enddo
      write(73) utdiff,vtdiff
      write(74) ttdiff,qtdiff,ptdiff
```

```
      read(53,rec=1)ut9,vt9
      read(54,rec=1)tt9,qt9,pt9
      endif
c
      enddo
      stop
      end
c
c
      subroutine trnv2x(ut,vt,tt,pt,qt,nx,lev,my,x)
c
c  Transfer variables to vector form
c     v1,v2,v3..... --------> x(v1,v2,v3.....)
c
c  INPUT: ut,vt,tt,pt,qt
c                              OUTPUT: x (Vector Form)
c
      dimension ut(nx,lev,my),vt(nx,lev,my),tt(nx,lev,my)
     >,         qt(nx,lev,my),pt(nx,my)
      real x(nx*lev*my*4+nx*my)
      do k=1,lev
      do j=1,my
      do i=1,nx
      idx_ut=i+(j-1)*nx+(k-1)*nx
      idx_vt=i+(j-1)*nx+(k-1)*nx+nx*lev*my
      idx_tt=i+(j-1)*nx+(k-1)*nx+nx*lev*my*2
      idx_pt=i+(j-1)*nx+nx*lev*my*3
      idx_qt=i+(j-1)*nx+(k-1)*nx+nx*lev*my*3+nx*my
      x(idx_ut)=ut(i,k,j)
      x(idx_vt)=vt(i,k,j)
      x(idx_tt)=tt(i,k,j)
      x(idx_pt)=pt(i,j)
      x(idx_qt)=qt(i,k,j)
      enddo
      enddo
      enddo
      return
      end
c
c
      subroutine Taylor_test(x99,x9,x,nx,lev,my,afa)
      real x(nx*lev*my*4+nx*my),x9(nx*lev*my*4+nx*my)
     >,    x99(nx*lev*my*4+nx*my)
      x99_x9_length=0.
      x_length=0.
      do idx=1,nx*lev*my*4+nx*my
        x99_x9_length=x99_x9_length+(x99(idx)-x9(idx))**2
        x_length=x_length+(afa*x(idx))**2
      enddo
      tay_test=sqrt(x99_x9_length)/(sqrt(x_length))
      write(26,10) afa,tay_test
   10 format(1x,'afa= ',f12.10,4x,'Taylor test= ',f20.13)
      return
      end
c
      subroutine Taylor_test_u(x99,x9,x,nx,lev,my,afa)
```



```
      real x(nx*lev*my*4+nx*my),x9(nx*lev*my*4+nx*my)
     >,    x99(nx*lev*my*4+nx*my)
      x99_x9_length=0.
      x_length=0.
      do k=1,lev
      do idx=1,nx*k*my
        x99_x9_length=x99_x9_length+(x99(idx)-x9(idx))**2
        x_length=x_length+(afa*x(idx))**2
      enddo
      tay_test=sqrt(x99_x9_length)/(sqrt(x_length))
      write(26,10) afa,k,tay_test
   10 format(1x,'afa= ',f12.10,4x,'k= ',i2,3x
     >,       'U comp. Taylor test= ',f20.13)
      x99_x9_length=0.
      x_length=0.
      enddo
      return
      end
c
c
      subroutine Taylor_test_v(x99,x9,x,nx,lev,my,afa)
      real x(nx*lev*my*4+nx*my),x9(nx*lev*my*4+nx*my)
     >,    x99(nx*lev*my*4+nx*my)
      x99_x9_length=0.
      x_length=0.
      do k=1,lev
      do idx=nx*lev*my+1,nx*my*k+(nx*lev*my)
        x99_x9_length=x99_x9_length+(x99(idx)-x9(idx))**2
        x_length=x_length+(afa*x(idx))**2
      enddo
      tay_test=sqrt(x99_x9_length)/(sqrt(x_length))
      write(26,10) afa,k,tay_test
   10 format(1x,'afa= ',f12.10,4x,'k= ',i2,3x
     >,       'V comp. Taylor test= ',f20.13)
      x99_x9_length=0.
      x_length=0.
      enddo
      return
      end
c
      subroutine Taylor_test_t(x99,x9,x,nx,lev,my,afa)
      real x(nx*lev*my*4+nx*my),x9(nx*lev*my*4+nx*my)
     >,    x99(nx*lev*my*4+nx*my)
      x99_x9_length=0.
      x_length=0.
      do k=1,lev
      do idx=nx*lev*my*2+1,nx*my*k+(nx*lev*my*2)
        x99_x9_length=x99_x9_length+(x99(idx)-x9(idx))**2
        x_length=x_length+(afa*x(idx))**2
      enddo
      tay_test=sqrt(x99_x9_length)/(sqrt(x_length))
      write(26,10) afa,k,tay_test
   10 format(1x,'afa= ',f12.10,4x,'k= ',i2,3x
     >,       'T comp. Taylor test= ',f20.13)
      x99_x9_length=0.
      x_length=0.
```

```
      enddo
      return
      end
c
      subroutine Taylor_test_q(x99,x9,x,nx,lev,my,afa)
      real x(nx*lev*my*4+nx*my),x9(nx*lev*my*4+nx*my)
     >,    x99(nx*lev*my*4+nx*my)
      idx_i=nx*lev*my*3+nx*my
      x99_x9_length=0.
      x_length=0.
      do k=1,lev
      do idx=idx_i+1,nx*my*k+idx_i
        x99_x9_length=x99_x9_length+(x99(idx)-x9(idx))**2
        x_length=x_length+(afa*x(idx))**2
      enddo
      tay_test=sqrt(x99_x9_length)/(sqrt(x_length))
      write(26,10) afa,k,tay_test
   10 format(1x,'afa= ',f12.10,4x,'k= ',i2,3x
     >,      'Q comp. Taylor test= ',f20.13)
      x99_x9_length=0.
      x_length=0.
      enddo
      return
      end
c
      subroutine Taylor_test_ps(x99,x9,x,nx,lev,my,afa)
      real x(nx*lev*my*4+nx*my),x9(nx*lev*my*4+nx*my)
     >,    x99(nx*lev*my*4+nx*my)
      x99_x9_length=0.
      x_length=0.
      do idx=nx*lev*my*3+1,nx*my*lev*3+nx*my
        x99_x9_length=x99_x9_length+(x99(idx)-x9(idx))**2
        x_length=x_length+(afa*x(idx))**2
      enddo
      tay_test=sqrt(x99_x9_length)/(sqrt(x_length))
      write(26,10) afa,tay_test
   10 format(1x,'afa= ',f12.10,4x,'Ps comp. Taylor test= ',f20.13)
      return
      end
c
c
```

```
      subroutine cons
c
c***********************************************************************
c  this subroutine defines several important constants and arrays
c  used by the spectral forecast model.  they include physical
c  parameters, model vertical structure, matrix operators used in
c  the semi-implicit algorthm, and polynomial arrays used in the
c  spherical harmonic transforms.
c***********************************************************************
c
      include '../include/param.h'
      include '../include/const.h'
      include '../include/fftcom.h'
c
      dimension pp(19)
c
      character*60 ifout,namlsts,crdate,cntrl,rfile,randat,
     1             rvorout,rdivout,ocards,momtlmtmp,mastlmtmp,
     1             momadjtmp,masadjtmp
c
      data pp/1000.,987.5,970.,950.,927.5,900.,867.5,825.,767.5
     1,       692.5,595.,480.,365.,265.,185.,125.,80.,42.5,0.01/
c
c new add DEC/26/96 at NCAR
c
      data tmean9/    229.25  ,  209.14,  198.15,  199.12,  214.05,
     1     217.82  ,  227.33  ,  236.96,  247.70,  258.23,  266.92,
     2     273.94  ,  279.79  ,  283.94,  287.66,  290.40,  291.83,
     3     292.08/
      data sig/0.000000,0.015947,0.039867,0.071761,0.111628,0.159468,
     1         0.215282,0.279070,0.350831,0.435382,0.527741,0.622923,
     2         0.715947,0.801827,0.875581,0.932226,0.966777,0.989369,
     3         1.000000/
c
      data cp/1004.24/, rad/6.371e6/, omega/7.292e-5/, grav/9.80616/
     *, taui/999.0/, taue/120.0/, tauo/6.0/, dt/900.0/, tfilt/0.02/
     *, ptop/0.1/, ptmeans9/600.0/, tmeans/lev*300./, ksgeo/0/
     *, hfilt/1.5e15/, ckf/1./, cka/0.025/, cks/0.25/, sigb/0.7/
     *, ptmean9/1000./,hday/1./
c
      data lsimpl/.true./, lzadv/.true./, yesdia/.true./
     *, hdiff/.true./, rstrt/.false./
c
      data namlsts/'../namlsts' /
      data cntrl/'../cntrl' /
      data rfile/'../rfile' /
      data ocards/'../ocards'/
c
      character*60 rvortlmout,rdivtlmout,rvoradjout,rdivadjout
      data rvortlmout/'../data/rvortlm.dat'/
      data rdivtlmout/'../data/rdivtlm.dat'/
      data rvoradjout/'../data/rvoradj.dat'/
      data rdivadjout/'../data/rdivadj.dat'/
c
      character*60 rvorout9,rdivout9
      data rvorout9/'../data/rvor9.dat'/
```



```
      data rdivout9/'../data/rdiv9.dat'/
c
      namelist /modlst/ ksgeo,ptop,tfilt,dt,taui,taue
     1                , tauo,tauo_tmp,lsimpl,lzadv,yesdia,hdiff,hday
     2                , ckf,cka,cks,sigb,tmpf_need
c
      namelist /filst/ ifout,namlsts,crdate,cntrl,rfile,randat,
     1                 rvortlmout,rdivtlmout,rvoradjout,rdivadjout,
     1                 momtlmtmp,mastlmtmp,momadjtmp,masadjtmp
c
      namelist /perlist/ addperturb
c
c      sig(1)=0.
c      sig(lev+1)=1.
c      do 90 k=1,lev
c        dsig(k)=1./lev
c 90   continue
c      do 100 k =1,lev-1
c      sig(k+1)= sig(k)+dsig(k)
c      print *,'k=',k+1,sig(k+1),dsig(k+1)
c  100 continue
c
c CLOSE FOR REAL DATA TEST at NCAR
c
c      do 90 k=1,lev+1
c        sig(lev-k+2)=(pp(k)-ptop)/(ptmean-ptop)
c      print *,sig(lev-k+2)
c 90   continue
      do 100 k = lev,1,-1
      dsig(k)= sig(k+1) - sig(k)
  100 continue
c
      capa= 1.0/3.5
      rgas= capa*cp
      pi  = 4.0*atan(1.0)
      radsq= rad*rad
c
      open (unit=12,file='../filist'
     1,        form='formatted')
      read (12,filst,end=110)
c
  110 continue
c
      print filst
c
c  read namelist for model parameters
c
      open (unit=1,file=namlsts,form='formatted')
c
      read (1,modlst,end=120)
c
  120 continue
c
      open(19,file='../perlst',form='formatted')
      read(19,perlist,end=111)
  111 continue
```

```
c
      open (unit=2,file=crdate,form='formatted')
c
      read (2,900) idtg
 900  format(i8)
c
      ckf=ckf/86400.
      cka=cka/86400.
      cks=cks/86400.
      hfilt=1.0/(24.*3600*hday*((jtrun*(jtrun+1))/radsq)**2)
c
      print modlst
c read file of output directives specifying desired output
c fields.
c
      open (unit=4,file=ocards,form='formatted')
c
      do 80 k=1,1000
      numout= k
      read (4,800,iostat=io)  outdir(numout)
  800 format (a12)
      if (io.ne.0)  go to 85
      if (outdir(numout).eq.'nomodata')  go to 85
   80 continue
   85 numout= numout-1
      print *,' cons outdir(1)=',outdir(1),'  numout= ',numout
c
c  build pointer arrays for locating zonal and total wavenumber
c  values in the one-dimensional spherical harmonic arrays.
c
      call sortml (jtrun,mlmax,msort,lsort,mlsort)
c
      do 150 ml = 1, mlmax
      rl = lsort(ml)
      rm = msort(ml)-1
      rlm= rl-1.0
      if (msort(ml).eq.1)  rm= 0.0
      if (lsort(ml).eq.1)  rlm= 0.0
      eps4(ml)= rl*rlm/radsq
      cim(ml)= rm
  150 continue
c
c  gaussian quadrature weights and latitudes
c
      one = 1.0
      onem= -one
      call gausl3 (my,onem,one,weight,sinl)
c
      my2= my/2
cdir$ ivdep
      do 180 j = 1, my2
      sinl(my+1-j)  = -sinl(j)
      weight(my+1-j)= weight(j)
      onocos(j)     = 1.0/(1.0-sinl(j)*sinl(j))
      onocos(my+1-j)= onocos(j)
      cosl(j)       = 1.0/sqrt(onocos(j))
```



```
      cosl(my+1-j)  = cosl(j)
  180 continue
c
c  define coriolis parameter for each latitude
c
      do 190 j=1,my
      cor(j)= 2.0*omega*sinl(j)
  190 continue
c
c  initialize ifax and trigs for rfftmlt routine
c
      call fftfax (nx,ifax,trigs)
c
c  define associated legendre polynomials and their derivatives
c
      call lgndr (my2,jtrun,mlmax,mlsort,sinl,poly,dpoly)
c
c
c open outfile(input) file
c
c      lenout=(mlmax*2*lev*3)*8
c      open(11,file=ifout,form='unformatted',access='direct',
c     1     recl=lenout,status='unknown')
c
      lenspec=lspec*8
c      open(41,file=rfile,form='unformatted',access='direct',
c     1     recl=lenspec,status='unknown')
c      open(42,file=cntrl,status='unknown')
      if(taui.eq.0.)then
        itaui=taui+0.001
c        write(42,'(2i8)')itaui
        rstrt=.false.
c      else
c        read(42,'(2i8)')itaui
c        rstrt=.true.
c        taui=float(itaui)
c        print *,'****** restart taui=',taui,' ********'
      endif
c
      nxlevmy8=nx*lev*my*8
      nxlevmy8_2=nx*lev*my*8*2
      nxlevmy8_3=nx*lev*my*8*2+nx*my*8
c      open(67,file=randat,form='unformatted',access='direct',
c     1     recl=nxlevmy8,status='unknown')
c
      open(22,file=rvortlmout,form='unformatted',access='direct',
     >     recl=nxlevmy8,status='unknown')
      open(23,file=rdivtlmout,form='unformatted',access='direct',
     >     recl=nxlevmy8,status='unknown')
      open(32,file=rvoradjout,form='unformatted',access='direct',
     >     recl=nxlevmy8,status='unknown')
      open(33,file=rdivadjout,form='unformatted',access='direct',
     >     recl=nxlevmy8,status='unknown')
c
      open(24,file=rvorout9,form='unformatted',access='direct',
     >     recl=nxlevmy8,status='unknown')
```

```
      open(25,file=rdivout9,form='unformatted',access='direct',
     >     recl=nxlevmy8,status='unknown')
      open(26,file='../data/tlmcheck.dat',form='formatted',
     >     status='unknown')
c
      open(51,file='../data/momtlmtmp',form='unformatted'
     >,    access='direct',recl=nxlevmy8_2,status='unknown')
      open(52,file='../data/mastlmtmp',form='unformatted'
     >,    access='direct',recl=nxlevmy8_3,status='unknown')
      open(61,file='../data/momadjtmp',form='unformatted'
     >,    access='direct',recl=nxlevmy8_2,status='unknown')
      open(62,file='../data/masadjtmp',form='unformatted'
     >,    access='direct',recl=nxlevmy8_3,status='unknown')
c
      open(53,file='../data/momtmp9',form='unformatted'
     >,    access='direct',recl=nxlevmy8_2,status='unknown')
      open(54,file='../data/mastmp9',form='unformatted'
     >,    access='direct',recl=nxlevmy8_3,status='unknown')
c
c CLOSED on DEC 1996 at NCAR
c
c set REAL data file's name
c
c      open(55,file='/tmp/chtseng/data101600/sigful.dmsdat',
c     >     form='unformatted',status='old')
c
c OUTPUT data file's name
c
c      open(61,file='/tmp/chtseng/dytst/data/divgout.ggdat',
c     >     form='unformatted',status='unknown')
c      open(62,file='/tmp/chtseng/dytst/data/dragout.ggdat',
c     >     form='unformatted',status='unknown')
c      open(63,file='/tmp/chtseng/dytst/data/geopout.ggdat',
c     >     form='unformatted',status='unknown')
c      open(64,file='/tmp/chtseng/dytst/data/out24.ggdat',
c     >     form='unformatted',status='unknown')
c      open(65,file='/tmp/chtseng/dytst/data/out2d.ggdat',
c     >     form='unformatted',status='unknown')
c      open(66,file='/tmp/chtseng/dytst/data/outsigso.ggdat',
c     >     form='unformatted',status='unknown')
c      open(67,file='/tmp/chtseng/dytst/data/shumout.ggdat',
c     >     form='unformatted',status='unknown')
c      open(68,file='/tmp/chtseng/dytst/data/sigfulo.ggdat',
c     >     form='unformatted',status='unknown')
c      open(69,file='/tmp/chtseng/dytst/data/surfout.ggdat',
c     >     form='unformatted',status='unknown')
c      open(70,file='/tmp/chtseng/dytst/data/tempout.ggdat',
c     >     form='unformatted',status='unknown')
c      open(71,file='/tmp/chtseng/dytst/data/vortout.ggdat',
c     >     form='unformatted',status='unknown')
c      open(72,file='/tmp/chtseng/dytst/data/windout.ggdat',
c     >     form='unformatted',status='unknown')
c
      return
      end
```

```
      subroutine gausl3 (n,xa,xb,wt,ab)
c
c weights and abscissas for nth order gaussian quadrature on (xa,xb).
c input arguments
c
c n  -the order desired
c xa -the left endpoint of the interval of integration
c xb -the right endpoint of the interval of integration
c output arguments
c ab -the n calculated abscissas
c wt -the n calculated weights
c
      implicit double precision (a-h,o-z)
c
      real  ab(n) ,wt(n),xa,xb
c
c machine dependent constants---
c  tol - convergence criterion for double precision iteration
c  pi  - given to 15 significant digits
c  c1  -  1/8                      these are coefficients in mcmahon"s
c  c2  -  -31/(2*3*8**2)           expansions of the kth zero of the
c  c3  -  3779/(2*3*5*8**3)        bessel function j0(x) (cf. abramowitz,
c  c4  -  -6277237/(3*5*7*8**5)    handbook of mathematical functions).
c  u   -  (1-(2/pi)**2)/4
c
      data tol/1.d-14/,pi/3.14159265358979/,u/.148678816357662/
      data c1,c2,c3,c4/.125,-.080729166666667,.246028645833333,
     1                 -1.82443876720609 /
c
c maximum number of iterations before giving up on convergence
c
      data maxit /5/
c
c arithmetic statement function for converting integer to double
c
      dbli(i) = dble(float(i))
c
      ddif = .5d0*(dble(xb)-dble(xa))
      dsum = .5d0*(dble(xb)+dble(xa))
      if (n .gt. 1) go to 101
      ab(1) = 0.
      wt(1) = 2.*ddif
      go to 107
  101 continue
      nnp1 = n*(n+1)
      cond = 1./sqrt((.5+float(n))**2+u)
      lim = n/2
c
      do 105 k=1,lim
         b = (float(k)-.25)*pi
         bisq = 1./(b*b)
c
c rootbf approximates the kth zero of the bessel function j0(x)
c
         rootbf = b*(1.+bisq*(c1+bisq*(c2+bisq*(c3+bisq*c4))))
c
```



```
c      initial guess for kth root of legendre poly p-sub-n(x)
c
         dzero = cos(rootbf*cond)
         do 103 i=1,maxit
c
            dpm2 = 1.d0
            dpm1 = dzero
c
c       recursion relation for legendre polynomials
c
            do 102 nn=2,n
               dp = (dbli(2*nn-1)*dzero*dpm1-dbli(nn-1)*dpm2)/dbli(nn)
               dpm2 = dpm1
               dpm1 = dp
  102       continue
            dtmp = 1.d0/(1.d0-dzero*dzero)
            dppr = dbli(n)*(dpm2-dzero*dp)*dtmp
            dp2pri = (2.d0*dzero*dppr-dbli(nnp1)*dp)*dtmp
            drat = dp/dppr
c
c       cubically-convergent iterative improvement of root
c
            dzeri = dzero-drat*(1.d0+drat*dp2pri/(2.d0*dppr))
            ddum= dabs(dzeri-dzero)
         if (ddum .le. tol) go to 104
            dzero = dzeri
  103    continue
         print 504
  504    format(1x,' in gausl3, convergence failed')
  104    continue
         ddifx = ddif*dzero
         ab(k) = dsum-ddifx
         wt(k) = 2.d0*(1.d0-dzero*dzero)/(dbli(n)*dpm2)**2*ddif
         i = n-k+1
         ab(i) = dsum+ddifx
         wt(i) = wt(k)
  105 continue
c
      if (mod(n,2) .eq. 0) go to 107
      ab(lim+1) = dsum
      nm1 = n-1
      dprod = n
      do 106 k=1,nm1,2
         dprod = dbli(nm1-k)*dprod/dbli(n-k)
  106 continue
      wt(lim+1) = 2.d0/dprod**2*ddif
  107 return
      end
c
c
      subroutine lgndr (my2,jtrun,mlmax,mlsort,sinl,poly,dpoly)
c
c  generate legendre polynomials and their derivatives on the
c  gaussian latitudes
c
c ***input***
```

```
c
c  my2:  number of gaussian latitudes from south pole and equator
c  jtrun:  zonal wavenumber truncation limit
c  mlmax: total number of triangular truncation spherical harmonics
c  mlsort: pointer array of 1-d indexs at functions of zonal and
c          total wavenumbers
c  sinl: sin of gaussian latitudes
c
c  ***output***
c
c  poly: associated legendre coefficients
c  dpoly: d(poly)/d(sinl)
c
c*****************************************************************
c
c ref= belousov, s. l., 1962= tables of normalized associated
c        legendre polynomials. pergamon press, new york
c
      dimension poly(mlmax,my2),dpoly(mlmax,my2),sinl(my2)
     *, mlsort(jtrun,jtrun)
c
      parameter (jtrunx= 100)
      dimension pnm(jtrunx+1,jtrunx),dpnm(jtrunx+1,jtrunx)
c
c sinl is sin(latitude) = cos(colatitude)
c pnm(np,mp) is legendre polynomial p(n,m) with np=n+1, mp=m+1
c pnm(mp,np+1) is x derivative of p(n,m) with np=n+1, mp=m+1
c
      jtrunp= jtrun+1
      do 1001 j=1,my2
      xx= sinl(j)
      sn= sqrt(1.0-xx*xx)
        sn2i = 1.0/(1.0 - xx*xx)
      rt2= sqrt(2.0)
        c1 = rt2
c
        pnm(1,1) = 1.0/rt2
      theta=-atan(xx/sqrt(1.0-xx*xx))+2.0*atan(1.0)
c
      do 20 n=1,jtrun
        np = n + 1
      fn=n
        fn2 = fn + fn
        fn2s = fn2*fn2
c eq 22
      c1= c1*sqrt(1.0-1.0/fn2s)
      c3= c1/sqrt(fn*(fn+1.0))
        ang = fn*theta
        s1 = 0.0
        s2 = 0.0
        c4 = 1.0
        c5 = fn
        a = -1.0
        b = 0.0
c
      do 27 kp=1,np,2
```



```
        k = kp - 1
      s2= s2+c5*sin(ang)*c4
      if (k.eq.n) c4 = 0.5*c4
      s1= s1+c4*cos(ang)
        a = a + 2.0
        b = b + 1.0
      fk=k
        ang = theta*(fn - fk - 2.0)
        c4 = (a*(fn - b + 1.0)/(b*(fn2 - a)))*c4
        c5 = c5 - 2.0
   27 continue
c eq 19
        pnm(np,1) = s1*c1
c eq 21
        pnm(np,2) = s2*c3
   20 continue
c
      do 4 mp=3,jtrunp
        m = mp - 1
      fm= m
        fm1 = fm - 1.0
        fm2 = fm - 2.0
        fm3 = fm - 3.0
      c6= sqrt(1.0+1.0/(fm+fm))
c eq 23
        pnm(mp,mp) = c6*sn*pnm(m,m)
      if (mp - jtrunp) 3,4,4
    3 continue
        nps = mp + 1
c
      do 41 np=nps,jtrunp
        n = np - 1
      fn= n
        fn2 = fn + fn
        c7 = (fn2 + 1.0)/(fn2 - 1.0)
        c8 = (fm1 + fn)/((fm + fn)*(fm2 + fn))
      c= sqrt((fn2+1.0)*c8*(fm3+fn)/(fn2-3.0))
      d= -sqrt(c7*c8*(fn-fm1))
      e= sqrt(c7*(fn-fm)/(fn+fm))
c eq 17
        pnm(np,mp) = c*pnm(np-2,mp-2)
     1             + xx*(d*pnm(np-1,mp-2) + e*pnm(np - 1,mp))
   41 continue
    4 continue
c
      do 50 mp=1,jtrun
      fm= mp-1.0
        fms = fm*fm
      do 50 np=mp,jtrun
      fnp= np
        fnp2 = fnp + fnp
        cf = (fnp*fnp - fms)*(fnp2 - 1.0)/(fnp2 + 1.0)
      cf= sqrt(cf)
c der
      dpnm(np,mp)   = -sn2i*(cf*pnm(np+1,mp) - fnp*xx*pnm(np,mp))
   50 continue
```

```
c
      do 71 m=1,jtrun
      do 71 l=m,jtrun
      ml= mlsort(m,l)
      poly(ml,j)= pnm(l,m)
      dpoly(ml,j)=dpnm(l,m)
   71 continue
      dpoly(1,j)= 0.0
 1001 continue
      return
      end
c
c
      subroutine invmtx (a,na,v,nv,n,d,ip,ier)
c
c
c dimension of          a(na,n),v(nv,n),ip(2*n)
c arguments
c
c
c purpose               invmtx calculates the inverse of the input
c                       matrix a using gaussian elimination with full
c                       pivoting.
c
c
c usage                 call invmtx (a,na,v,nv,n,d,ip,ier)
c
c arguments
c
c on input              a
c                         a two-dimensional variable with row dimension
c                         na and column dimension n.  on input, a
c                         contains the elements of the matrix to be
c                         inverted.
c
c                       na
c                         an integer input variable set equal to the
c                         row dimension of a as declared in the
c                         dimension statement of the calling program.
c
c                       nv
c                         an integer input variable set equal to the
c                         row dimension of v as declared in the
c                         dimension statement of the calling program.
c
c                       n
c                         an integer input variable set equal to the
c                         column dimension of a.
c
c                       ip
c                         an integer array used internally for working
c                         storage.  it must have dimension at least
c                         2*n.
c
c on output             v
c                         a two-dimensional variable with row dimension
```



```
c                         nv and column dimension n.  on output, v
c                         contains the inverse of a.
c                              if a is entered twice in the parameter
c                              list, replacing v, then on output the
c                              array a will contain the inverse matrix,
c                              and the original a will be destroyed.
c
c                       d
c                         a real variable which on output contains the
c                         determinant of a.
c
c                       ier
c                         an integer error flag.
c                           = 33  if the matrix a is singular.
c                           = 0  otherwise.
c
c entry points          invmtx
c
c common blocks         none
c
c i/o                   if the
c                       matrix a has a zero pivot element (i.e., a is
c                       singular), the message
c                          *matrix singular in invmtx*
c                       is printed.
c
c precision             single
c
c language              fortran
c
c
c
      implicit real (a-h,o-z)
c
c
      real  a(na,n)    ,v(nv,n)    ,d
      dimension ip(1)
c
      ier = 0
c
c store a in v
c
      do 102 j=1,n
         ip (j) = 0
         do 101 i=1,n
            v(i,j) = a(i,j)
  101    continue
  102 continue
      d = 1.
      do 111 m=1,n
         vmax = 0.
         do 107 j=1,n
            if (ip(j)) 107,103,107
c
c find maximum pivot element
c
```

```
  103       do 106 i=1,n
                if (ip(i)) 106,104,106
  104          vh = abs(v(i,j))
                if (vmax-vh) 105,106,106
  105          vmax = vh
                k = i
                l = j
  106       continue
  107    continue
c
       ip(l) = k
       ip(n+m) = l
       d = d*v(k,l)
       pvt = v(k,l)
       if (pvt .eq. 0.) go to 114
       v(k,l) = 1.
       do 108 j=1,n
          hold = v(k,j)
          v(k,j) = v(l,j)
          v(l,j) = hold/pvt
  108    continue
       do 110 i=1,n
          if (i .eq. l) go to 110
          hold = v(i,l)
          v(i,l) = 0.
          do 109 j=1,n
             v(i,j) = v(i,j)-v(l,j)*hold
  109       continue
  110    continue
  111 continue
c
c permute final inverse matrix
c
      do 113 j=1,n
         m = n-j+1
         l = ip(n+m)
         k = ip(l)
         if (k .eq. l) go to 113
         d = -d
         do 112 i=1,n
            hold = v(i,l)
            v(i,l) = v(i,k)
            v(i,k) = hold
  112    continue
  113 continue
      return
  114 ier = 33
      print 888
  888 format(1x,'matrix singualr in invmtx')
      return
      end
c
c
      subroutine mtxmlp (a,b,c,lm)
c
c  matrix multiply routine
```



```
c
c  ***input***
c
c  a:  first array in matrix product
c  b:  second array in matrix product
c  lm: order of matrices
c
c ***output***
c
c  c: matrix product
c
c ************************************************************
c
      dimension a(lm,lm),b(lm,lm),c(lm,lm)
      call zilch(c,lm*lm)
      do 1 k=1,lm
      do 1 j=1,lm
      do 1 i=1,lm
    1 c(i,j)= c(i,j)+a(i,k)*b(k,j)
      return
      end
c
c
      subroutine sortml (jtrun,mlmax,msort,lsort,mlsort)
c
c  sortml builds pointer arrays for functional dependency between
c  1-d spectral index and zonal and total wavenumber indices.  this
c  subroutine reflects the coefficient storage strategy used in the
c  model
c
c ***input***
c
c  jtrun:  zonal and total wavenumber limit
c  mlmax:  total number of 1-d spectral index for triangular trunc
c
c  ***output***
c
c  msort:  zonal wavenumber as function of 1-d spectral index
c  lsort:  total wavenumber as function of 1-d spectral index
c  mlsort: total wavenumber index as function of zonal and total
c          wavenumber
c
c ****************************************************************
c
      dimension msort(mlmax),lsort(mlmax),mlsort(jtrun,jtrun)
c
      mlx= (jtrun/2)*((jtrun+1)/2)
      ml= 0
      do 1 k=1,jtrun-1,2
      do 1 m=1,jtrun-k
      ml= ml+1
      mlp= ml+mlx
      mlsort(m,m+k)= mlp
      mlsort(m,m+k-1)= ml
      msort(ml)= m
      lsort(ml)= m+k-1
```

```
      msort(mlp)= m
      lsort(mlp)= m+k
    1 continue
c
      ml= mlp
      do 2 m=2,jtrun,2
      ml= ml+1
      mlsort(m,jtrun)= ml
      msort(ml)= m
      lsort(ml)= jtrun
    2 continue
      return
      end
c
c
      subroutine trandv (jtrun,mlmax,nx,my,ll,ut,vt,w,cim
     *, onocos,poly,dpoly,vor,div)
c
c  subroutine to do grid point velocities to spectral vorticity
c  and divergence
c
c **** const ****
c
c  jtrun: zonal wavenumber truncation limit
c  mlmax: total number of spectral coefficients (horizontal field)
c  nx: e-w dimension no.
c  my: n-w dimension no.
c  ll: number of vertical levels to be transformed
c  w: gaussian quadrature weights
c  cim: zonal wavenumber array
c  onocos: 1.o/(cos(lat)**2)
c  poly: legendre polynomials
c  dpoly: d(poly)/d(sin(lat))
c
c *** input variables ***
c
c  ut: e-w velocity component
c  vt: n-s velocity component
c
c *** output variables ***
c
c  vor: spectral vorticity
c  div: spectral divergence
c
c ******************************************
c
      dimension poly(mlmax,my/2),dpoly(mlmax,my/2),cim(mlmax)
     *, onocos(my),w(my),ut(nx,ll,my),vt(nx,ll,my),vor(mlmax,2,ll)
     *, div(mlmax,2,ll)
c
      include '../include/fftcom.h'
csun  include '../include/paramt.h' .. change im,jm,mlm to nx,my,mlmax
      dimension cc(nx+3,my),dd(nx+3,my)
      dimension work(nx*my,2),cfac(mlmax),dfac(mlmax)
c
      mlx= (jtrun/2)*((jtrun+1)/2)
```



```
c
      do 30 k=1,ll
      do 23 j=1,my
cdir@ ivdep
      do 23 i=1,nx
      cc(i,j)= ut(i,k,j)
      dd(i,j)= vt(i,k,j)
   23 continue
c
      call rfftmlt(cc,work,trigs,ifax,1,nx+3,nx,my,-1)
      call rfftmlt(dd,work,trigs,ifax,1,nx+3,nx,my,-1)
c
c  to begin quadrature integral we compute contribution without
c  adding to an existing sum
c
cdir@ ivdep
      do 82 ml=1,mlmax
      cfac(ml)= w(1)*dpoly(ml,1)
      dfac(ml)= w(1)*onocos(1)*cim(ml)*poly(ml,1)
   82 continue
c
      m1= 0
      do 62 l=jtrun-1,1,-2
cdir@ ivdep
         do 63 m = 1, l
         mm= 2*m-1
         mp= mm+1
         ml= m+m1
         mk= ml+mlx
            vor(ml,1,k)=  cfac(ml)*(cc(mm,1)
     *     -cc(mm,my))-dfac(ml)*(dd(mp,1)+dd(mp,my))
c
            vor(mk,1,k)=  cfac(mk)*(cc(mm,1)
     *     +cc(mm,my))-dfac(mk)*(dd(mp,1)-dd(mp,my))
c
            vor(ml,2,k)=  cfac(ml)*(cc(mp,1)
     *     -cc(mp,my))+dfac(ml)*(dd(mm,1)+dd(mm,my))
c
            vor(mk,2,k)=  cfac(mk)*(cc(mp,1)
     *     +cc(mp,my))+dfac(mk)*(dd(mm,1)-dd(mm,my))
c
            div(ml,1,k)=- cfac(ml)*(dd(mm,1)
     *     -dd(mm,my))-dfac(ml)*(cc(mp,1)+cc(mp,my))
c
            div(mk,1,k)=- cfac(mk)*(dd(mm,1)
     *     +dd(mm,my))-dfac(mk)*(cc(mp,1)-cc(mp,my))
c
            div(ml,2,k)=- cfac(ml)*(dd(mp,1)
     *     -dd(mp,my))+dfac(ml)*(cc(mm,1)+cc(mm,my))
c
            div(mk,2,k)=- cfac(mk)*(dd(mp,1)
     *     +dd(mp,my))+dfac(mk)*(cc(mm,1)-cc(mm,my))
c
   63    continue
         m1= m1+l
   62 continue
```

```
c
      ml= mlx*2
cdir@ ivdep
      do 64 m=2,jtrun,2
      ml=ml+1
      mm= 2*m-1
      mp= mm+1
            vor(ml,1,k)=  cfac(ml)*(cc(mm,1)
     *     -cc(mm,my))-dfac(ml)*(dd(mp,1)+dd(mp,my))
c
            vor(ml,2,k)=  cfac(ml)*(cc(mp,1)
     *     -cc(mp,my))+dfac(ml)*(dd(mm,1)+dd(mm,my))
c
            div(ml,1,k)=- cfac(ml)*(dd(mm,1)
     *     -dd(mm,my))-dfac(ml)*(cc(mp,1)+cc(mp,my))
c
            div(ml,2,k)=- cfac(ml)*(dd(mp,1)
     *     -dd(mp,my))+dfac(ml)*(cc(mm,1)+cc(mm,my))
c
   64 continue
c
c  now do rest of guassian latitudes by adding to accumulating
c  sum
c
      do 70 j=2,my/2
c
cdir@ ivdep
      do 84 ml=1,mlmax
      cfac(ml)= w(j)*dpoly(ml,j)
      dfac(ml)= w(j)*onocos(j)*cim(ml)*poly(ml,j)
   84 continue
c
      jj= my-j+1
      m1= 0
      do 72 l=jtrun-1,1,-2
cdir@ ivdep
         do 73 m = 1, l
         mm= 2*m-1
         mp= mm+1
         ml= m+m1
         mk= ml+mlx
            vor(ml,1,k)= vor(ml,1,k)+cfac(ml)*(cc(mm,j)
     *     -cc(mm,jj))-dfac(ml)*(dd(mp,j)+dd(mp,jj))
c
            vor(mk,1,k)= vor(mk,1,k)+cfac(mk)*(cc(mm,j)
     *     +cc(mm,jj))-dfac(mk)*(dd(mp,j)-dd(mp,jj))
c
            vor(ml,2,k)= vor(ml,2,k)+cfac(ml)*(cc(mp,j)
     *     -cc(mp,jj))+dfac(ml)*(dd(mm,j)+dd(mm,jj))
c
            vor(mk,2,k)= vor(mk,2,k)+cfac(mk)*(cc(mp,j)
     *     +cc(mp,jj))+dfac(mk)*(dd(mm,j)-dd(mm,jj))
c
            div(ml,1,k)= div(ml,1,k)-cfac(ml)*(dd(mm,j)
```



```
            div(mk,1,k)= div(mk,1,k)-cfac(mk)*(dd(mm,j)
     *     +dd(mm,jj))-dfac(mk)*(cc(mp,j)-cc(mp,jj))
c
            div(ml,2,k)= div(ml,2,k)-cfac(ml)*(dd(mp,j)
     *     -dd(mp,jj))+dfac(ml)*(cc(mm,j)+cc(mm,jj))
c
            div(mk,2,k)= div(mk,2,k)-cfac(mk)*(dd(mp,j)
     *     +dd(mp,jj))+dfac(mk)*(cc(mm,j)-cc(mm,jj))
c
   73    continue
         m1= m1+l
   72 continue
c
      ml= mlx*2
cdir@ ivdep
      do 65 m=2,jtrun,2
      ml=ml+1
      mm= 2*m-1
      mp= mm+1
            vor(ml,1,k)= vor(ml,1,k)+cfac(ml)*(cc(mm,j)
     *     -cc(mm,jj))-dfac(ml)*(dd(mp,j)+dd(mp,jj))
c
            vor(ml,2,k)= vor(ml,2,k)+cfac(ml)*(cc(mp,j)
     *     -cc(mp,jj))+dfac(ml)*(dd(mm,j)+dd(mm,jj))
c
            div(ml,1,k)= div(ml,1,k)-cfac(ml)*(dd(mm,j)
     *     -dd(mm,jj))-dfac(ml)*(cc(mp,j)+cc(mp,jj))
c
            div(ml,2,k)= div(ml,2,k)-cfac(ml)*(dd(mp,j)
     *     -dd(mp,jj))+dfac(ml)*(cc(mm,j)+cc(mm,jj))
c
   65 continue
   70 continue
   30 continue
c
      return
      end
c
c
      subroutine adtrandv (jtrun,mlmax,nx,my,ll,ut,vt,w,cim
     *, onocos,poly,dpoly,vor,div)
c
c  adjoint of trandv.f
c  subroutine to do grid point velocities to spectral vorticity
c  and divergence
c
c **** const ****
c
c  jtrun: zonal wavenumber truncation limit
c  mlmax: total number of spectral coefficients (horizontal field)
c  nx: e-w dimension no.
c  my: n-w dimension no.
c  ll: number of vertical levels to be transformed
c  w: gaussian quadrature weights
```

```
c  poly: legendre polynomials
c  dpoly: d(poly)/d(sin(lat))
c
c *** output variables ***
c
c  ut: e-w velocity component
c  vt: n-s velocity component
c
c *** input ***
c
c  vor: spectral vorticity
c  div: spectral divergence
c
c ******************************************
c
      dimension poly(mlmax,my/2),dpoly(mlmax,my/2),cim(mlmax)
     *, onocos(my),w(my),ut(nx,ll,my),vt(nx,ll,my),vor(mlmax,2,ll)
     *, div(mlmax,2,ll)
c
      include '../include/fftcom.h'
csun  include '../include/paramt.h' .. change im,jm,mlm to nx,my,mlmax
      dimension cc(nx+3,my),dd(nx+3,my),RECIP(NX)
      dimension work(nx*my,2),cfac(mlmax),dfac(mlmax)
c
c Define coefficient of adjoint rfftmlt
c
      RECIP(1)=1.E0/float(NX)
      RECIP(2)=RECIP(1)
      DO 1009 I=3,NX
      RECIP(I)=1.E0/float(2*NX)
1009  CONTINUE
c
      mlx= (jtrun/2)*((jtrun+1)/2)
c
c ===================================================================
c  Begin adjoint code
c ===================================================================
c
      do 30 k=1,ll
      do j=1,my
      do i=1,nx+3
      cc(i,j)=0.
      dd(i,j)=0.
      enddo
      enddo
c
c  now do rest of guassian latitudes by adding to accumulating
c  sum
c
      do 70 j=2,my/2
cdir@ ivdep
      do 84 ml=1,mlmax
      cfac(ml)= w(j)*dpoly(ml,j)
      dfac(ml)= w(j)*onocos(j)*cim(ml)*poly(ml,j)
   84 continue
      jj= my-j+1
```



```
c
      ml=mlx*2
cdir@ ivdep
      do 65 m=2,jtrun,2
      ml=ml+1
      mm= 2*m-1
      mp= mm+1
c           vor(ml,1,k)= vor(ml,1,k)+cfac(ml)*(cc(mm,j)
c     *     -cc(mm,jj))-dfac(ml)*(dd(mp,j)+dd(mp,jj))
            cc(mm,j)=cfac(ml)*vor(ml,1,k)+cc(mm,j)
            cc(mm,jj)=-cfac(ml)*vor(ml,1,k)+cc(mm,jj)
            dd(mp,j)=-dfac(ml)*vor(ml,1,k)+dd(mp,j)
            dd(mp,jj)=-dfac(ml)*vor(ml,1,k)+dd(mp,jj)
c
c           vor(ml,2,k)= vor(ml,2,k)+cfac(ml)*(cc(mp,j)
c     *     -cc(mp,jj))+dfac(ml)*(dd(mm,j)+dd(mm,jj))
            cc(mp,j)=cfac(ml)*vor(ml,2,k)+cc(mp,j)
            cc(mp,jj)=-cfac(ml)*vor(ml,2,k)+cc(mp,jj)
            dd(mm,j)=dfac(ml)*vor(ml,2,k)+dd(mm,j)
            dd(mm,jj)=dfac(ml)*vor(ml,2,k)+dd(mm,jj)
c
c           div(ml,1,k)= div(ml,1,k)-cfac(ml)*(dd(mm,j)
c     *     -dd(mm,jj))-dfac(ml)*(cc(mp,j)+cc(mp,jj))
            dd(mm,j)=-cfac(ml)*div(ml,1,k)+dd(mm,j)
            dd(mm,jj)=cfac(ml)*div(ml,1,k)+dd(mm,jj)
            cc(mp,j)=-dfac(ml)*div(ml,1,k)+cc(mp,j)
            cc(mp,jj)=-dfac(ml)*div(ml,1,k)+cc(mp,jj)
c
c           div(ml,2,k)= div(ml,2,k)-cfac(ml)*(dd(mp,j)
c     *     -dd(mp,jj))+dfac(ml)*(cc(mm,j)+cc(mm,jj))
            dd(mp,j)=-cfac(ml)*div(ml,2,k)+dd(mp,j)
            dd(mp,jj)=cfac(ml)*div(ml,2,k)+dd(mp,jj)
            cc(mm,j)=dfac(ml)*div(ml,2,k)+cc(mm,j)
            cc(mm,jj)=dfac(ml)*div(ml,2,k)+cc(mm,jj)
c
   65 continue
c
c
      m1= 0
      do 72 l=jtrun-1,1,-2
cdir@ ivdep
         do 73 m = 1, l
         mm= 2*m-1
         mp= mm+1
         ml= m+m1
         mk= ml+mlx
c           vor(ml,1,k)= vor(ml,1,k)+cfac(ml)*(cc(mm,j)
c     *     -cc(mm,jj))-dfac(ml)*(dd(mp,j)+dd(mp,jj))
            cc(mm,j)=cfac(ml)*vor(ml,1,k)+cc(mm,j)
            cc(mm,jj)=-cfac(ml)*vor(ml,1,k)+cc(mm,jj)
            dd(mp,j)=-dfac(ml)*vor(ml,1,k)+dd(mp,j)
            dd(mp,jj)=-dfac(ml)*vor(ml,1,k)+dd(mp,jj)
c
c           vor(mk,1,k)= vor(mk,1,k)+cfac(mk)*(cc(mm,j)
c     *     +cc(mm,jj))-dfac(mk)*(dd(mp,j)-dd(mp,jj))
            cc(mm,j)=cfac(mk)*vor(mk,1,k)+cc(mm,j)
```

```
            cc(mm,jj)=cfac(mk)*vor(mk,1,k)+cc(mm,jj)
            dd(mp,j)=-dfac(mk)*vor(mk,1,k)+dd(mp,j)
            dd(mp,jj)=dfac(mk)*vor(mk,1,k)+dd(mp,jj)
c
c            vor(ml,2,k)= vor(ml,2,k)+cfac(ml)*(cc(mp,j)
c     *     -cc(mp,jj))+dfac(ml)*(dd(mm,j)+dd(mm,jj))
            cc(mp,j)=cfac(ml)*vor(ml,2,k)+cc(mp,j)
            cc(mp,jj)=-cfac(ml)*vor(ml,2,k)+cc(mp,jj)
            dd(mm,j)=dfac(ml)*vor(ml,2,k)+dd(mm,j)
            dd(mm,jj)=dfac(ml)*vor(ml,2,k)+dd(mm,jj)
c
c            vor(mk,2,k)= vor(mk,2,k)+cfac(mk)*(cc(mp,j)
c     *     +cc(mp,jj))+dfac(mk)*(dd(mm,j)-dd(mm,jj))
            cc(mp,j)=cfac(mk)*vor(mk,2,k)+cc(mp,j)
            cc(mp,jj)= cfac(mk)*vor(mk,2,k)+cc(mp,jj)
            dd(mm,j)=dfac(mk)*vor(mk,2,k)+dd(mm,j)
            dd(mm,jj)=-dfac(mk)*vor(mk,2,k)+dd(mm,jj)
c
c            div(ml,1,k)= div(ml,1,k)-cfac(ml)*(dd(mm,j)
c     *     -dd(mm,jj))-dfac(ml)*(cc(mp,j)+cc(mp,jj))
            dd(mm,j)=-cfac(ml)*div(ml,1,k)+dd(mm,j)
            dd(mm,jj)=cfac(ml)*div(ml,1,k)+dd(mm,jj)
            cc(mp,j)=-dfac(ml)*div(ml,1,k)+cc(mp,j)
            cc(mp,jj)=-dfac(ml)*div(ml,1,k)+cc(mp,jj)
c
c            div(mk,1,k)= div(mk,1,k)-cfac(mk)*(dd(mm,j)
c     *     +dd(mm,jj))-dfac(mk)*(cc(mp,j)-cc(mp,jj))
            dd(mm,j)=-cfac(mk)*div(mk,1,k)+dd(mm,j)
            dd(mm,jj)=-cfac(mk)*div(mk,1,k)+dd(mm,jj)
            cc(mp,j)=-dfac(mk)*div(mk,1,k)+cc(mp,j)
            cc(mp,jj)=dfac(mk)*div(mk,1,k)+cc(mp,jj)
c
c            div(ml,2,k)= div(ml,2,k)-cfac(ml)*(dd(mp,j)
c     *     -dd(mp,jj))+dfac(ml)*(cc(mm,j)+cc(mm,jj))
            dd(mp,j)=-cfac(ml)*div(ml,2,k)+dd(mp,j)
            dd(mp,jj)=cfac(ml)*div(ml,2,k)+dd(mp,jj)
            cc(mm,j)=dfac(ml)*div(ml,2,k)+cc(mm,j)
            cc(mm,jj)=dfac(ml)*div(ml,2,k)+cc(mm,jj)
c
c            div(mk,2,k)= div(mk,2,k)-cfac(mk)*(dd(mp,j)
c     *     +dd(mp,jj))+dfac(mk)*(cc(mm,j)-cc(mm,jj))
            dd(mp,j)=-cfac(mk)*div(mk,2,k)+dd(mp,j)
            dd(mp,jj)=-cfac(mk)*dd(mp,jj)+dd(mp,jj)
            cc(mm,j)=dfac(mk)*div(mk,2,k)+cc(mm,j)
            cc(mm,jj)=-dfac(mk)*div(mk,2,k)+cc(mm,jj)
c
   73    continue
         m1= m1+l
   72 continue
c
   70 continue
c
c-----------------------------------------------------------------
c  Redefine const
c
      do 82 ml=1,mlmax
```



```
      cfac(ml)= w(1)*dpoly(ml,1)
      dfac(ml)= w(1)*onocos(1)*cim(ml)*poly(ml,1)
   82 continue
c-----------------------------------------------------------------
c
c
c  to begin quadrature integral we compute contribution without
c  adding to an existing sum
c
      ml= mlx*2
cdir@ ivdep
      do 64 m=2,jtrun,2
      ml=ml+1
      mm= 2*m-1
      mp= mm+1
c            vor(ml,1,k)=  cfac(ml)*(cc(mm,1)
c     *     -cc(mm,my))-dfac(ml)*(dd(mp,1)+dd(mp,my))
            cc(mm,1)=cfac(ml)*vor(ml,1,k)+cc(mm,1)
            cc(mm,my)=-cfac(ml)*vor(ml,1,k)+cc(mm,my)
            dd(mp,1)=-dfac(ml)*vor(ml,1,k)+dd(mp,1)
            dd(mp,my)=-dfac(ml)*vor(ml,1,k)+dd(mp,my)
c
c            vor(ml,2,k)=  cfac(ml)*(cc(mp,1)
c     *     -cc(mp,my))+dfac(ml)*(dd(mm,1)+dd(mm,my))
            cc(mp,1)=cfac(ml)*vor(ml,2,k)+cc(mp,1)
            cc(mp,my)=cfac(ml)*vor(ml,2,k)+cc(mp,my)
            dd(mm,1)=dfac(ml)*vor(ml,2,k)+dd(mm,1)
            dd(mm,my)=dfac(ml)*vor(ml,2,k)+dd(mm,my)
c
c            div(ml,1,k)=- cfac(ml)*(dd(mm,1)
c     *     -dd(mm,my))-dfac(ml)*(cc(mp,1)+cc(mp,my))
            dd(mm,1)=-cfac(ml)*div(ml,1,k)+dd(mm,1)
            dd(mm,my)=cfac(ml)*div(ml,1,k)+dd(mm,my)
            cc(mp,1)=-dfac(ml)*div(ml,1,k)+cc(mp,1)
            cc(mp,my)=-dfac(ml)*div(ml,1,k)+cc(mp,my)
c
c            div(ml,2,k)=- cfac(ml)*(dd(mp,1)
c     *     -dd(mp,my))+dfac(ml)*(cc(mm,1)+cc(mm,my))
            dd(mp,1)=-cfac(ml)*div(ml,2,k)+dd(mp,1)
            dd(mp,my)=cfac(ml)*div(ml,2,k)+dd(mp,my)
            cc(mm,1)=dfac(ml)*div(ml,2,k)+cc(mm,1)
            cc(mm,my)=dfac(ml)*div(ml,2,k)+cc(mm,my)
c
   64 continue
c
      m1= 0
      do 62 l=jtrun-1,1,-2
cdir@ ivdep
         do 63 m = 1, l
         mm= 2*m-1
         mp= mm+1
         ml= m+m1
         mk= ml+mlx
c            vor(ml,1,k)=  cfac(ml)*(cc(mm,1)
c     *     -cc(mm,my))-dfac(ml)*(dd(mp,1)+dd(mp,my))
            cc(mm,1)=cfac(ml)*vor(ml,1,k)+cc(mm,1)
```

```
            cc(mm,my)=-cfac(ml)*vor(ml,1,k)+cc(mm,my)
            dd(mp,1)=-dfac(ml)*vor(ml,1,k)+dd(mp,1)
            dd(mp,my)=-dfac(ml)*vor(ml,1,k)+dd(mp,my)
c
c            vor(mk,1,k)=  cfac(mk)*(cc(mm,1)
c     *     +cc(mm,my))-dfac(mk)*(dd(mp,1)-dd(mp,my))
            cc(mm,1)=cfac(mk)*vor(mk,1,k)+cc(mm,1)
            cc(mm,my)=cfac(mk)*vor(mk,1,k)+cc(mm,my)
            dd(mp,1)=-dfac(mk)*vor(mk,1,k)+dd(mp,1)
            dd(mp,my)=dfac(mk)*vor(mk,1,k)+dd(mp,my)
c
c            vor(ml,2,k)=  cfac(ml)*(cc(mp,1)
c     *     -cc(mp,my))+dfac(ml)*(dd(mm,1)+dd(mm,my))
            cc(mp,1)=cfac(ml)*vor(ml,2,k)+cc(mp,1)
            cc(mp,my)=-cfac(ml)*vor(ml,2,k)+cc(mp,my)
            dd(mm,1)=dfac(ml)*vor(ml,2,k)+dd(mm,1)
            dd(mm,my)=dfac(ml)*vor(ml,2,k)+dd(mm,my)
c
c            vor(mk,2,k)=  cfac(mk)*(cc(mp,1)
c     *     +cc(mp,my))+dfac(mk)*(dd(mm,1)-dd(mm,my))
            cc(mp,1)=cfac(mk)*vor(mk,2,k)+cc(mp,1)
            cc(mp,my)=cfac(mk)*vor(mk,2,k)+cc(mp,my)
            dd(mm,1)=dfac(mk)*vor(mk,2,k)+dd(mm,1)
            dd(mm,my)=-dfac(mk)*vor(mk,2,k)+dd(mm,my)
c
c            div(ml,1,k)=- cfac(ml)*(dd(mm,1)
c     *     -dd(mm,my))-dfac(ml)*(cc(mp,1)+cc(mp,my))
            dd(mm,1)=-cfac(ml)*div(ml,1,k)+dd(mm,1)
            dd(mm,my)=cfac(ml)*div(ml,1,k)+dd(mm,my)
            cc(mp,1)=-dfac(ml)*div(ml,1,k)+cc(mp,1)
            cc(mp,my)=-dfac(ml)*div(ml,1,k)+cc(mp,my)
c
c            div(mk,1,k)=- cfac(mk)*(dd(mm,1)
c     *     +dd(mm,my))-dfac(mk)*(cc(mp,1)-cc(mp,my))
            dd(mm,1)=-cfac(mk)*div(mk,1,k)+dd(mm,1)
            dd(mm,my)=-cfac(mk)*div(mk,1,k)+dd(mm,my)
            cc(mp,1)=-dfac(mk)*div(mk,1,k)+cc(mp,1)
            cc(mp,my)=dfac(mk)*div(mk,1,k)+cc(mp,my)
c
c            div(ml,2,k)=- cfac(ml)*(dd(mp,1)
c     *     -dd(mp,my))+dfac(ml)*(cc(mm,1)+cc(mm,my))
            dd(mp,1)=-cfac(ml)*div(ml,2,k)+dd(mp,1)
            dd(mp,my)=cfac(ml)*div(ml,2,k)+dd(mp,my)
            cc(mm,1)=dfac(ml)*div(ml,2,k)+cc(mm,1)
            cc(mm,my)=dfac(ml)*div(ml,2,k)+cc(mm,my)
c
c            div(mk,2,k)=- cfac(mk)*(dd(mp,1)
c     *     +dd(mp,my))+dfac(mk)*(cc(mm,1)-cc(mm,my))
            dd(mp,1)=-cfac(mk)*div(mk,2,k)+dd(mp,1)
            dd(mp,my)=-cfac(mk)*div(mk,2,k)+dd(mp,my)
            cc(mm,1)=dfac(mk)*div(mk,2,k)+cc(mm,1)
            cc(mm,my)=-dfac(mk)*div(mk,2,k)+cc(mm,my)
c
   63    continue
         m1= m1+l
   62 continue
```



```
c
c adjoint of the RFFTMLT (negative)
c
cWORK(NX*MY,2)
      DO I=1,NX*MY
        WORK(I,1)=0.
        WORK(I,2)=0.
      ENDDO
c
      DO J=1,MY
      DO I=1,NX
         CC(I,J)=CC(I,J)*RECIP(I)
      ENDDO
      ENDDO
      call rfftmlt(cc,work,trigs,ifax,1,nx+3,nx,my,1)
c
      DO I=1,NX*MY
        WORK(I,1)=0.
        WORK(I,2)=0.
      ENDDO
c
      DO J=1,MY
      DO I=1,NX
         DD(I,J)=DD(I,J)*RECIP(I)
      ENDDO
      ENDDO
      call rfftmlt(dd,work,trigs,ifax,1,nx+3,nx,my,1)
c
      do 23 j=1,my
cdir$ ivdep
      do 23 i=1,nx
      ut(i,k,j)=cc(i,j)+ut(i,k,j)
      vt(i,k,j)=dd(i,j)+vt(i,k,j)
   23 continue
c
   30 continue
c
      return
      end
c
c
      subroutine tranrs (jtrun,mlmax,nx,my,ll,poly,w,r,s)
c
c  subroutine to transform a scalar grid point field to spectral
c  coefficients
c
c *** const ***
c
c  jtrun: zonal wavenumber truncation limit
c  mlmax: total number of spherical harmonic coeff  (horizontal field)
c  nx: e-w dimension no.
c  my: n-s dimension no.
c  ll: number of vertical levels to transform
c  poly: legendre polynomials
c  w: gaussian quadrature weights
c
```

```
c *** input variable ***
c
c  r: 3-dim input grid pt. field to be transformed
c
c *** output variables ***
c
c  s: spectral coefficient fields
c
c  *********************************
c
      dimension poly(mlmax,my/2),s(mlmax,2,ll),r(nx,ll,my),w(my)
      include '../include/fftcom.h'
csun  include '../include/paramt.h' .. change im,jm to nx,my
      dimension cc(nx+3,my),work(nx*my,2)
c
      mlx= (jtrun/2)*((jtrun+1)/2)
c
      do 30 k=1,ll
c
c  put grid point fields into two dimensional horizontal array
c
      do 23 j=1,my
      do 23 i=1,nx
      cc(i,j)= r(i,k,j)
   23 continue
c
c  fft for each guassian latitude of 2-d field
c
      call rfftmlt(cc,work,trigs,ifax,1,nx+3,nx,my,-1)
c
c  to start quadrature integral we compute contribution without
c  adding to existing sum
c
      m1= 0
      do 62 l=jtrun-1,1,-2
cdir@    ivdep
         do 63 m = 1, l
         mm= 2*m-1
         mp= mm+1
         ml= m+m1
         mk= ml+mlx
            s(ml,1,k) = w(1)*poly(ml,1)*(cc(mm,1)+cc(mm,my))
            s(ml,2,k) = w(1)*poly(ml,1)*(cc(mp,1)+cc(mp,my))
            s(mk,1,k) = w(1)*poly(mk,1)*(cc(mm,1)-cc(mm,my))
            s(mk,2,k) = w(1)*poly(mk,1)*(cc(mp,1)-cc(mp,my))
   63    continue
      m1= m1+l
   62 continue
c
      ml= mlx*2
cdir@ ivdep
      do 64 m=2,jtrun,2
      ml=ml+1
      mm= 2*m-1
      mp= mm+1
      s(ml,1,k)= w(1)*poly(ml,1)*(cc(mm,1)+cc(mm,my))
```



```
      s(ml,2,k)= w(1)*poly(ml,1)*(cc(mp,1)+cc(mp,my))
   64 continue
c
c  for rest of quadrature integral we add contribution to
c  existing sum
c
      do 70 j=2,my/2
      jj= my-j+1
      m1= 0
      do 72 l=jtrun-1,1,-2
cdir@    ivdep
         do 73 m = 1, l
         mm= 2*m-1
         mp= mm+1
         ml= m+m1
         mk= ml+mlx
            s(ml,1,k) = s(ml,1,k)+w(j)*poly(ml,j)*(cc(mm,j)+cc(mm,jj))
            s(ml,2,k) = s(ml,2,k)+w(j)*poly(ml,j)*(cc(mp,j)+cc(mp,jj))
            s(mk,1,k) = s(mk,1,k)+w(j)*poly(mk,j)*(cc(mm,j)-cc(mm,jj))
            s(mk,2,k) = s(mk,2,k)+w(j)*poly(mk,j)*(cc(mp,j)-cc(mp,jj))
   73    continue
      m1= m1+l
   72 continue
c
      ml= mlx*2
cdir@ ivdep
      do 65 m=2,jtrun,2
      ml=ml+1
      mm= 2*m-1
      mp= mm+1
      s(ml,1,k)= s(ml,1,k)+w(j)*poly(ml,j)*(cc(mm,j)+cc(mm,jj))
      s(ml,2,k)= s(ml,2,k)+w(j)*poly(ml,j)*(cc(mp,j)+cc(mp,jj))
   65 continue
   70 continue
   30 continue
c
      return
      end
c
c
      subroutine adtranrs (jtrun,mlmax,nx,my,ll,poly,w,r,s)
c
c  adjoint of tranrs.f
c  subroutine to transform a spectral coefficient field to
c  grid point form
c
c *** input const ***
c
c  jtrun: zonal wavenumber resolution limit
c  mlmax: number of spectral coefficients (horizontal field)
c  nx: e-w dimension no.
c  my: n-s dimension no.
c  ll: number of levels to transform
c  poly: legendre polynomials
c  w: Gaussian quadrature weight
c
```

```
c *** input variable ***
c
c  s: spectral coefficient array to transform
c
c *** output variables ***
c
c  r: 3-d output grid point fields
c
c  ************************************
c
      dimension poly(mlmax,my/2),s(mlmax,2,ll),r(nx,ll,my),w(my)
      dimension RECIP(NX)
      include '../include/fftcom.h'
csun  include '../include/paramt.h' .. change im,jm to nx,my
      dimension cc(nx+3,my),work(nx*my,2)
c
      mlx= (jtrun/2)*((jtrun+1)/2)
c
      do 30 k=1,ll
      do j=1,my
      do i=1,nx+3
      cc(i,j)=0.
      enddo
      enddo
c
c  for rest of quadrature integral we add contribution to
c  existing sum
c
      do 70 j=2,my/2
      ml= mlx*2
      jj= my-j+1
      m1= 0
      do 65 m=2,jtrun,2
      ml=ml+1
      mm= 2*m-1
      mp= mm+1
c      s(ml,1,k)= s(ml,1,k)+w(j)*poly(ml,j)*(cc(mm,j)+cc(mm,jj))
c      s(ml,2,k)= s(ml,2,k)+w(j)*poly(ml,j)*(cc(mp,j)+cc(mp,jj))
      cc(mm,j)=w(j)*poly(ml,j)*s(ml,1,k)+cc(mm,j)
      cc(mm,jj)=w(j)*poly(ml,j)*s(ml,1,k)+cc(mm,jj)
      cc(mp,j)=w(j)*poly(ml,j)*s(ml,2,k)+cc(mp,j)
      cc(mp,jj)=w(j)*poly(ml,j)*s(ml,2,k)+cc(mp,jj)
   65 continue
c
      ml= mlx*2
      jj= my-j+1
      m1= 0
      do 72 l=jtrun-1,1,-2
c
         do 73 m = 1, l
         mm= 2*m-1
         mp= mm+1
         ml= m+m1
         mk= ml+mlx
c            s(ml,1,k) = s(ml,1,k)+w(j)*poly(ml,j)*(cc(mm,j)+cc(mm,jj))
c            s(ml,2,k) = s(ml,2,k)+w(j)*poly(ml,j)*(cc(mp,j)+cc(mp,jj))
```



```
c            s(mk,1,k) = s(mk,1,k)+w(j)*poly(mk,j)*(cc(mm,j)-cc(mm,jj))
c            s(mk,2,k) = s(mk,2,k)+w(j)*poly(mk,j)*(cc(mp,j)-cc(mp,jj))
            cc(mm,j)=w(j)*poly(ml,j)*s(ml,1,k)+cc(mm,j)
            cc(mm,jj)=w(j)*poly(ml,j)*s(ml,1,k)+cc(mm,jj)
            cc(mp,j)=w(j)*poly(ml,j)*s(ml,2,k)+cc(mp,j)
            cc(mp,jj)=w(j)*poly(ml,j)*s(ml,2,k)+cc(mp,jj)
            cc(mm,j)=w(j)*poly(mk,j)*s(mk,1,k)+cc(mm,j)
            cc(mm,jj)=-1.*w(j)*poly(mk,j)*s(mk,1,k)+cc(mm,jj)
            cc(mp,j)=w(j)*poly(mk,j)*s(mk,2,k)+cc(mp,j)
            cc(mp,jj)=-1.*w(j)*poly(mk,j)*s(mk,2,k)+cc(mp,jj)
   73    continue
         m1= m1+l
   72 continue
   70 continue
c
      ml= mlx*2
cdir@ ivdep
      do 64 m=2,jtrun,2
      ml=ml+1
      mm= 2*m-1
      mp= mm+1
c      s(ml,1,k)= w(1)*poly(ml,1)*(cc(mm,1)+cc(mm,my))
c      s(ml,2,k)= w(1)*poly(ml,1)*(cc(mp,1)+cc(mp,my))
      cc(mm,1)=w(1)*poly(ml,1)*s(ml,1,k)+cc(mm,1)
      cc(mm,my)=w(1)*poly(ml,1)*s(ml,1,k)+cc(mm,my)
      cc(mp,1)=w(1)*poly(ml,1)*s(ml,2,k)+cc(mp,1)
      cc(mp,my)=w(1)*poly(ml,1)*s(ml,2,k)+cc(mp,my)
   64 continue
c
c  to start quadrature integral we compute contribution without
c  adding to existing sum
c
      m1= 0
      do 62 l=jtrun-1,1,-2
cdir@    ivdep
         do 63 m = 1, l
         mm= 2*m-1
         mp= mm+1
         ml= m+m1
         mk= ml+mlx
c            s(ml,1,k) = w(1)*poly(ml,1)*(cc(mm,1)+cc(mm,my))
c            s(ml,2,k) = w(1)*poly(ml,1)*(cc(mp,1)+cc(mp,my))
c            s(mk,1,k) = w(1)*poly(mk,1)*(cc(mm,1)-cc(mm,my))
c            s(mk,2,k) = w(1)*poly(mk,1)*(cc(mp,1)-cc(mp,my))
            cc(mm,1)=w(1)*poly(ml,1)*s(ml,1,k)+cc(mm,1)
            cc(mm,my)=w(1)*poly(ml,1)*s(ml,1,k)+cc(mm,my)
            cc(mp,1)=w(1)*poly(ml,1)*s(ml,2,k)+cc(mp,1)
            cc(mp,my)=w(1)*poly(ml,1)*s(ml,2,k)+cc(mp,my)
            cc(mm,1)=w(1)*poly(mk,1)*s(mk,1,k)+cc(mm,1)
            cc(mm,my)=-1.*w(1)*poly(mk,1)*s(mk,1,k)+cc(mm,my)
            cc(mp,1)=w(1)*poly(mk,1)*s(mk,2,k)+cc(mp,1)
            cc(mp,my)=-1.*w(1)*poly(mk,1)*s(mk,2,k)+cc(mp,my)
   63    continue
         m1= m1+l
   62 continue
c
```

```
      program test_adj
c
c Global ForeCaST Dry(X) adjoint Model
c
      include '../include/param.h'
      include '../include/const.h'
      include '../include/gridtlm.h'
      include '../include/gridtlm9.h'
      include '../include/spec.h'
      include '../include/spec9.h'
c
      character*1 bas_tmpf_need
c
      dimension ut(nx,lev,my),vt(nx,lev,my),tt(nx,lev,my)
     >,         pt(nx,my),qt(nx,lev,my)
      dimension ut9(nx,lev,my),vt9(nx,lev,my),tt9(nx,lev,my)
     >,         pt9(nx,my),qt9(nx,lev,my)
c     dimension vornow(mlmax,2,lev),divnow(mlmax,2,lev)
c     >,         temnow(mlmax,2,lev),qnow(mlmax,2,lev)
c     >,         plnow(mlmax,2)
      real coeff(mlmax*2*lev*4+mlmax*2)
      dimension xx(nx*lev*my*4+nx*my),axx(nx*lev*my*4+nx*my)
c     >,         yy(mlmax*2*lev*4+mlmax*2)
     >,        yy(nx*lev*my*4+nx*my)
c     >,         yy(lev*2)
      real tbar(lev),qbar(lev)
c
c
c  logical io units:
c
c  1:  input namelist file = 'namlsts'
c  3:  spectrum coeff output file='bspt??'
c 12:  filist
c
c
c  get model constants
c
      call cons
c
c  read in initial data and prepare for initialization/forecast
c
c  set up Basic state
c
      call grossby(nx,my,lev,sinl,tmean9,ut9,vt9,tt9,qt9,pt9
     *,            sig,ptop,pk9,pk29,plt9,tref9,capa,cosl)
c
c  set up perturb field
c
      call perturb(nx,my,lev,sinl,tmean9,tmean
     *,            pt9,pk9,pk29,tt9
     *,            ut,vt,tt,qt,pt,sig,ptop
     *,            pk,pk2,plt,tref,capa,cosl)
c
      print *,'finished setting initial field'
c     do j=1,my
c     do k=1,nx
```



```
c     do i=1,nx
c     ut(i,k,j)=0.
c     vt(i,k,j)=0.
c     tt(i,k,j)=0.
c     qt(i,k,j)=1.
c     pt(i,j)=0.
c     enddo
c     enddo
c     enddo
c
c  Inner product test
c
c     call trans(1,ut,vt,tt,pt,qt,nx,my,lev,xx,idx)
c     do i=1,idx
c       axx(i)=xx(i)
c     enddo
      n3d=nx*lev*my
      n=0
      do j=1,my
      do k=1,lev
      do i=1,nx
      n=n+1
      axx(n)=ut(i,k,j)
      axx(n+n3d)=vt(i,k,j)
      axx(n+2*n3d)=tt(i,k,j)
      axx(n+3*n3d)=qt(i,k,j)
      enddo
      enddo
      enddo
      n=0
      do j=1,my
      do i=1,nx
      n=n+1
      axx(n+4*n3d)=pt(i,j)
      enddo
      enddo
c
c  time integration
c  INPUT: ut,vt,tt,pt,qt
c                                  OUTPUT: ut,vt,tt,pt,qt
c
      bas_tmpf_need='y'
      call Mintgrt(ut9,vt9,tt9,pt9,qt9,bas_tmpf_need)
      call Lintgrt(ut,vt,tt,pt,qt,coeff)
c
c     call trans(1,ut,vt,tt,pt,qt,nx,my,lev,yy(1),idx)
c     n=0
c     do j=1,my
c     do k=1,lev
c     do i=1,nx
c     n=n+1
c     yy(n)=ut(i,k,j)
c     yy(n+n3d)=vt(i,k,j)
c     yy(n+2*n3d)=tt(i,k,j)
c     yy(n+3*n3d)=qt(i,k,j)
c     enddo
```

```
      dimension vorten(mlmax,2,lev),divten(mlmax,2,lev)
     >,         temten(mlmax,2,lev),qten(mlmax,2,lev)
     >,         plten(mlmax,2)
      real yy(mlmax*2*lev*4+mlmax*2)
c
      print *,'vorten(10,1,5)= ',vorten(10,1,5)
      idy=mlmax*2*lev*4+mlmax*2
      mlmax2=mlmax*2
      do k=1,lev
      do i=1,mlmax2
      idy_vor=i+(k-1)*mlmax2
      idy_div=i+(k-1)*mlmax2+mlmax2*lev
      idy_tem=i+(k-1)*mlmax2+mlmax2*lev*2
      idy_qte=i+(k-1)*mlmax2+mlmax2*lev*3
      idy_plt=i+mlmax2*lev*4
      yy(idy_vor)=vorten(i,1,k)
      yy(idy_div)=divten(i,1,k)
      yy(idy_tem)=temten(i,1,k)
      yy(idy_qte)=qten(i,1,k)
      yy(idy_plt)=plten(i,1)
      enddo
      enddo
      return
      end
c
c
      subroutine trnv2x_out_2(tbar,qbar,lev,yy,idy)
      dimension tbar(lev),qbar(lev)
      real yy(lev*2)
c
      idy=lev*2
      do k=1,lev
      idy_tbar=k
      idy_qbar=k+lev
      yy(idy_tbar)=tbar(k)
      yy(idy_qbar)=qbar(k)
      enddo
      return
      end
c
c
      subroutine trnv2x_out(ut,vt,tt,pt,qt,nx,my,lev,yy,idy)
c
      dimension ut(nx,lev,my),vt(nx,lev,my),pt(nx,my)
     *,         tt(nx,lev,my),qt(nx,lev,my)
      real yy(nx*lev*my*4+nx*my)
c
      idy=nx*lev*my*4+nx*my
      do j=1,my
      do k=1,lev
      do i=1,nx
      idy_ut=i+(k-1)*nx+(j-1)*nx
      idy_vt=i+(k-1)*nx+(j-1)*nx+nx*lev*my
      idy_tt=i+(k-1)*nx+(j-1)*nx+nx*lev*my*2
      idy_qt=i+(k-1)*nx+(j-1)*nx+nx*lev*my*3
      idy_pt=i+(j-1)*nx+nx*lev*my*4
```



```
      yy(idy_ut)=ut(i,k,j)
      yy(idy_vt)=vt(i,k,j)
      yy(idy_tt)=tt(i,k,j)
      yy(idy_qt)=qt(i,k,j)
      yy(idy_pt)=pt(i,j)
      enddo
      enddo
      enddo
      return
      end
c
c
      subroutine trnv2x(ut,vt,tt,pt,qt,nx,lev,my,x)
c
c  Transfer variables to vector form
c     v1,v2,v3..... --------> x(v1,v2,v3.....)
c
c  INPUT: ut,vt,tt,pt,qt
c                              OUTPUT: x (Vector Form)
c
      dimension ut(nx,lev,my),vt(nx,lev,my),tt(nx,lev,my)
     >,         qt(nx,lev,my),pt(nx,my)
      real x(nx*lev*my*4+nx*my)
      do k=1,lev
      do j=1,my
      do i=1,nx
      idx_ut=i+(j-1)*nx+(k-1)*nx
      idx_vt=i+(j-1)*nx+(k-1)*nx+nx*lev*my
      idx_tt=i+(j-1)*nx+(k-1)*nx+nx*lev*my*2
      idx_pt=i+(j-1)*nx+nx*lev*my*3
      idx_qt=i+(j-1)*nx+(k-1)*nx+nx*lev*my*3+nx*my
      x(idx_ut)=ut(i,k,j)
      x(idx_vt)=vt(i,k,j)
      x(idx_tt)=tt(i,k,j)
      x(idx_pt)=pt(i,j)
      x(idx_qt)=qt(i,k,j)
      enddo
      enddo
      enddo
      return
      end
c
c
```