GRAS SAF Report 03
Ref: SAF/GRAS/METO/REP/GSR/003
Web: www.grassaf.org
Date: 20 November 2007

The EUMETSAT
Network of
Satellite Application
Facilities

GRAS SAF
GRAS Meteorology

GRAS SAF Report 03

ROPP Minimiser - minROPP

Huw Lewis

Met Office, UK

Document Author Table

	Name	Function	Date	Comments
Prepared by:	H. Lewis	GRAS SAF Project Team	20 November 2007	
Reviewed by:	D. Offiler	GRAS SAF Project Team	20 November 2007	
Approved by:	K.B. Lauritsen	GRAS SAF Project Manager	17 October 2008	

Document Change Record

Issue/Revision	Date	By	Description
Version 1.0	20 Nov 2007	HL	First version

GRAS SAF Project

The GRAS SAF is a EUMETSAT-funded project responsible for operational processing of GRAS radio occultation data from the Metop satellites. The GRAS SAF delivers bending angle, refractivity, temperature, pressure, and humidity profiles in near-real time and offline for NWP and climate users. The offline profiles are further processed into climate products consisting of gridded monthly zonal means of bending angle, refractivity, temperature, humidity, and geopotential heights together with error descriptions.

The GRAS SAF also maintains the Radio Occultation Processing Package (ROPP) which contains software modules that will aid users wishing to process, quality-control and assimilate radio occultation data from any radio occultation mission into NWP and other models.

The GRAS SAF Leading Entity is the Danish Meteorological Institute (DMI), with Cooperating Entities: i) European Centre for Medium-Range Weather Forecasts (ECMWF) in Reading, United Kingdom, ii) Institut D'Estudis Espacials de Catalunya (IEEC) in Barcelona, Spain, and iii) Met Office in Exeter, United Kingdom. To get access to our products or to read more about the project please go to http://www.grassaf.org.

1 Background

The ROPP 1dVar retrieval aims to provide profiles of pressure, temperature and humidity using the refractivity or bending angle profile measured from a GPS radio occultation. This is achieved in the `ropp_1dvar_cost` subroutine through the minimisation of a quadratic cost function J. This routine calculates

$$J = \frac{1}{2}\langle x - x_b | B^{-1} | x - x_b \rangle + \frac{1}{2}\langle y_o - H(x) | O^{-1} | y_o - H(x) \rangle \tag{1.1}$$

where the background state x_b is given by the state vector, y_o is the observation vector, H is the forward model and B and O are the background and the observation covariance matrices respectively.

The minimisation of the cost function J is performed in the subroutine `ropp_1dvar_solve` by calling an external minimiser. In ROPP v1.0 this is conducted using `M1QN3`, one of the INRIA limited memory Quasi-Newton codes designed to solve large-scale unconstrained minimisation problems (1). The software is written in Fortran 77. `M1QN3` is implemented in ROPP in reverse communication mode, so that the cost function itself is calculated by the `ropp_1dvar_cost` subroutine called from `ropp_1dvar_solve` before proceeding to the minimisation problem.

The `M1QN3` code is not directly available for download from the INRIA website. but requires a licence application form to be completed and returned to the authors. The `M1QN3` licence agreement specifies that

> “The software is to be used with an academic or research purpose only. In particular, it will not be used for commercial applications or in production codes.”

To avoid potential licencing issues arising, especially in view of the requirement for ROPP software to be used for operational purposes, an ROPP-specific minimiser is required. This development also removes the need for each user to acquire a licence and copy of the code.

A new minimiser `ropp_1dvar_minropp` has been written in Fortran 90 for implementation in the ROPP 1dVar retrieval. It is intended that this routine will replace the requirement for `M1QN3` in future release versions of ROPP. `minROPP` has been developed as part of the GRAS-SAF from a new Quasi-Newton code written within the Met Office directly from the open literature (2). Modifications to that Met Office code have been required for ROPP, specifically to replicate the reverse communication mode utilised in ROPP which was not directly available in the original Met Office code.

This document provides an overview of `minROPP`, and demonstrates its performance compared with `M1QN3`.

2 Limited-memory quasi-Newton method

Both `M1QN3` and `minROPP` apply a limited-memory quasi-Newton method to minimise the cost function. Further details are provided by (2) and (3). An overview is provided to aid understanding of the code if required.

At each iteration k a new value of the state vector x is obtained as

$$x_{k+1} = x_k - \alpha_k G_k^{-1} \nabla J(x_k) \tag{2.1}$$

where $g_k \equiv \nabla J$ is the gradient of the cost function, α_k is a step length and G_k is termed the Hessian, defined as $G_k = \nabla^2 J(x_k)$. In order to avoid the expense of computing G_k^{-1} at each iteration, the quasi-Newton method is formed as

$$x_{k+1} = x_k - \alpha_k W_k g_k \tag{2.2}$$

where matrix W_k is an approximation of the Hessian which satisfies the relation

$$W_{k+1}(g_{k+1} - g_k) = x_{k+1} - x_k \tag{2.3}$$

The product $p_k = W_k g_k$ specifies the search direction of the minimisation.

2.1 Preconditioning

The initial value for the direction vector p_k is set using Fletcher's scaling.

$$p_k = -g_k \left(2 + \frac{\delta J}{||g_k||^2} \right) \tag{2.4}$$

where δJ is the expected decrease of the cost function, computed in subroutine `ropp_1dvar_solve` as

$$\delta J = MAX[J_0 - 0.5 N_{obs}, 0.1 J_0] \tag{2.5}$$

for N_{obs} number of observations.

2.2 Nocedal's method

`M1QN3` and `minROPP` apply Nocedal's BFGS method to obtain W_{k+1} at each iteration (4). The matrix W_{k+1} is obtained by performing m updates of W_k.

$$W_{k+1} = W_k^m \tag{2.6}$$

At each update i,

$$W_k^{i+1} = \overline{BFGS}(W_k^i, y_{k-m+i}, s_{k-m+i}) \qquad \text{for } 0 \leq i \leq m-1 \tag{2.7}$$

The function $\overline{BFGS}(W, y, s)$ is given by

$$\overline{BFGS}(W_k, y_k, s_k) = \left(I - \frac{s_k \times y_k}{\langle y_k, s_k \rangle}\right) W_k \left(I - \frac{y_k \times s_k}{\langle y_k, s_k \rangle}\right) + \frac{s_k \times s_k}{\langle y_k, s_k \rangle} \tag{2.8}$$

where $y_k \equiv g_{k+1} - g_k$ and $s_k \equiv x_{k+1} - x_k$.

2.3 Diagonal scaling

At each iteration it is efficient to set $W_k^0 = D_k$ where D_k is a diagonal matrix. The ith diagonal element of matrix D is found using

$$D_{k+1}^{(i)} = \left(\frac{\langle D_k y_k, y_k \rangle}{\langle y_k, s_k \rangle D_k^{(i)}} + \frac{\langle y_k, e_i \rangle^2}{\langle y_k, s_k \rangle} - \frac{\langle D_k y_k, y_k \rangle \langle s_k, e_i \rangle^2}{\langle y_k, s_k \rangle \langle D_k^{-1} s_k, s_k \rangle (D_k^{(i)})^2} \right)^{-1} \tag{2.9}$$

The initial value of D is given by

$$D_0 = \frac{y_0^T s_0}{|y_0|^2} \tag{2.10}$$

Note that in `ropp_1dvar_minropp` the initial value of D is simply initialised to unity.

2.4 Step length

The step length α_k used at each iteration is determined by satisfying the Wolfe conditions

$$J(x_k + \alpha_k p_k) \leq J(0) + \rho \alpha_k \nabla J(0)^T p_k \tag{2.11}$$

$$\nabla J(x_k + \alpha_k p_k)^T p_k \geq \sigma \nabla J(0)^T p_k \tag{2.12}$$

where $\rho = 1 \times 10^{-4}$ and $\sigma = 0.9$ are pre-defined constants. Note that in `ropp_1dvar_minropp` the step size α is simply set equal to unity (2).

2.5 Reverse communication

The `M1QN3` algorithm is implemented in ROPP v1.0 with a reverse communication protocol. At each minimisation loop within `ropp_1dvar_solve`, a call to `M1QN3` is made and performs a single iteration to update the state vector x. The cost function J and its gradient ∇J are then re-evaluated by calling the subroutine `ropp_1dvar_cost`. This process continues until convergence is achieved. The `ropp_1dvar_minropp` routine has been developed from the Met Office code (2) in order to replicate this implementation. The logic of `ropp_1dvar_solve` is therefore unchanged with the introduction of the `minROPP` minimiser.

2.6 Convergence criteria

It is considered that convergence has been obtained at x_k in `M1QN3` if the ratio of the gradient of the cost function at x_k to the initial value is less than a pre-defined factor `epsg`. i.e. if

$$\frac{||g_k||}{||g_1||} < epsg \tag{2.13}$$

The value of `epsg` is set in the ROPP 1dVar configuration file (`epsg`=1×10^{-8}). This stopping criterion is also checked in `minROPP`.

Additional checks for convergence are performed in `ropp_1dvar_cost`. Convergence is assumed if either the state vector does not change by more than a set value between iterations,

$$|x_k - x_{k-1}|/\sqrt{B} < \texttt{max\_delta\_state} \tag{2.14}$$

where B is the background error covariance matrix, or the cost function does not change by more than a set value between iterations,

$$|J_k - J_{k-1}| < \texttt{max\_delta\_J} \tag{2.15}$$

These conditions need to be met for at least `conv_check_n_previous` successive iterations for convergence to be assumed. Parameters `max_delta_state` (0.1), `max_delta_J` (0.1) and `conv_check_n_previous` (2) are set in the ROPP 1dVar configuration file.

3 Minimiser implementation

3.1 `M1QN3` implementation

The `m1qn3` minimiser routine is currently implemented in ROPP `1`DVar (v1.0) with reverse communication and run in diagonal initial scaling (DIS) mode. With a reverse communication protocol, `m1qn3` is called at each loop and asked to perform a single iteration. The cost function and its gradient is then computed by a call to `ropp_1dvar_cost` until either `m_indic` or `c_indic` (return value from the cost function routine) is set to `0` following one of the convergence criteria being satisfied. For further details see (3).

Table 3.1 lists the arguments required by `M1QN3`. When operated in reverse communication mode subroutine `simul_rc` is an empty routine, since the cost function and its gradient are computed externally to `M1QN3`.

Users who wish to continue using the `M1QN3` minimiser, subject to the `M1QN3` license agreement, in ROPP to solve the `1`dVar problem may call subroutine `ropp_1dvar_solve_m1qn3`. It is envisaged that support for this routine and `M1QN3` will be withdrawn in future, and use of the new ROPP minimiser is strongly recommended.

3.2 `minrOPP` implementation

The new `minROPP` minimiser is to be implemented in future versions of the ROPP `1`dVar. This is called from subroutine `ropp_1dvar_solve`. The code is written to perform in an equivalent operation to the reverse communication mode of `M1QN3` with diagonal initial scaling. `ropp_1dvar_minropp` is called at each loop and asked to perform a single iteration. The cost function and its gradient is computed by `ropp_1dvar_cost` until either `m_indic` or `c_indic` (return value from the cost function routine) is set to `0` following one of the convergence criteria being satisfied. Table 3.2 lists the arguments required by `ropp_1dvar_minropp`.

3.2.1 Code organisation

Figure 3.1 illustrates the logic of the `minROPP` code. The dependence of the code logic on the `indic` communication flags is highlighted. On the first minimisation loop, `indic=1`, the search vector is initialised using Fletcher's scaling and the next value of the state vector x_{k+1} is computed in the linesearch routine. At this stage, `indic` is set to `4`, indicating that the data should be stored and the cost function and its gradient should be recalculated on exiting `ropp_1dvar_minropp`. On the next implementation with `indic=4`, the linesearch routine only resets `indic` back to `1` so that a new search vector is computed by Nocedal's method using the current and previous values of x and ∇J. After the new search vector

p_k has been computed the state vector x_{k+1} is determined and the routine again exits with `indic=4`.

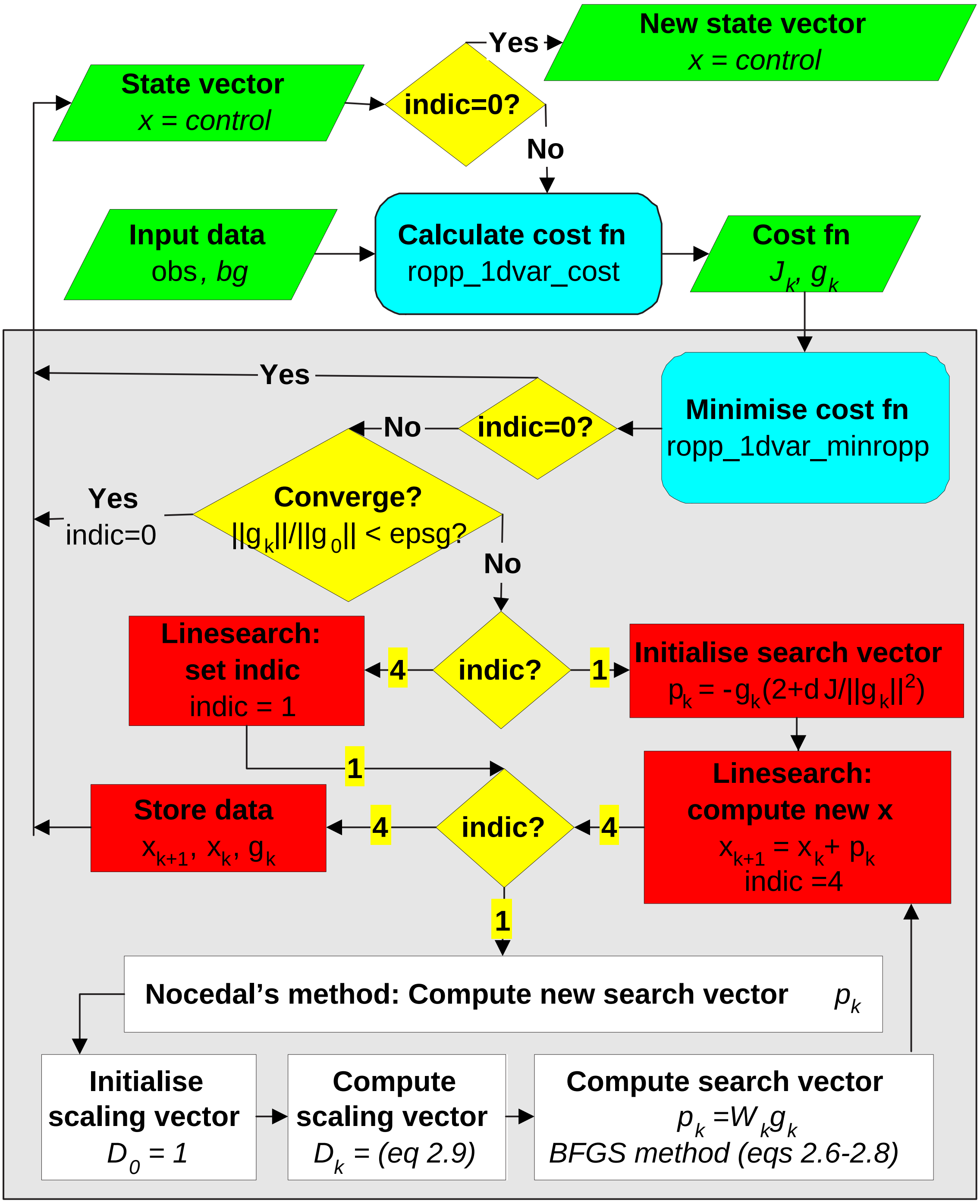

Figure 3.1: Sketch of `ropp_1dvar_minropp` minimisation routine

Name	Type	Description	ROPP v1.0 variable name	Initialised?
`simul`		subroutine to compute J, ∇J	`simul_rc`	external routine provided with m1qn3
`prosca`		subroutine to compute inner product	`euclid`	external routine provided with m1qn3
`contb`		subroutine to change vector basis	`ctonbe`	external routine provided with m1qn3
`ctcab`		subroutine to perform reverse of `contb`	`ctcabe`	external routine provided with m1qn3
`n`	int	dimension of problem	`size(control%state)`	
`x`	dbl arr	entry: initial value x_k, rtn: calculated x_{k+1}	`control%state`	preconditioning
`f`	dbl	value of J at x_k	`J`	call to calculate cost fn
`g`	dbl arr	value of ∇J at x_k	`J_grad`	call to calculate cost fn
`dxmin`	dbl	resolution of x	`config%m1qn3%dxmin`	set in config file (10^{-16})
`dfl`	dbl	expected decrease of J	`dJ`	computed in code
`epsg`	dbl	precision of stop criterion	`config%m1qn3%eps_grad`	set in config file (10^{-8})
`impres`	int	controls output	`config%m1qn3%impres`	set in config (to screen)
`io`	int	channel number for output	`unit`	set in code based on `impres`
`imode`	int arr	set running mode (imode(1))	`imode`	`config%m1qn3%imode` (DIS)
		set starting mode (imode(2))		set to 0 (cold start)
`omode`	int	specify output mode	`omode`	set to 0 (stop when `indic`=0)
`niter`	int	entry: max no. of iterations	`config%m1qn3%n_iter`	set in config file (1500)
		rtn: number of iterations		
`nsim`	int	entry: max no. of simulations	`config%m1qn3%n_simul`	set in config file (2500)
		rtn: number of simulations		
`iz`	int arr	address working array	`iwork`	not initialised
`dz`	dbl arr	address of working array	`work`	not initialised
`ndz`	int	dimension of working area	`n_work`	computed in code
`reverse`	logical	specify direct or reverse	`rev_com`	equal 'true' in code (reverse)
`indic`	int	flag monitoring communication	`m_indic`	set to 1 in code (1st iteration)
`izs`	int arr	address of working array	0	
`rzs`	real arr	address of working array	0.0	
`dzs`	dbl arr	address of working array	0.d0	

Table 3.1: Variables called by `M1QN3` routine in ROPP 1DVar

Name	Type	Description	ROPP variable name	Initialised?
`x`	dbl arr	entry: initial value x_k, rtn: calculated x_{k+1}	`control%state`	preconditioning
`J`	dbl	value of J at x_k	`J`	call to calculate cost fn
`g`	dbl arr	value of ∇J at x_k	`J_grad`	call to `ropp_1dvar_cost`
`p`	dbl arr	search direction vector at x_k	`J_dir`	call to `ropp_1dvar_cost`
`dJ`	dbl	expected decrease of J	`dJ`	computed in code
`gconv`	dbl	precision of stop criterion	`gconv`	computed from `epsg` in code
`niter`	int	number of iterations	`n_iter`	
`indic`	int	flag monitoring communication	`m_indic`	set to 1 in code (1st iteration)
`miter`	int	maximum no. of iterations	`config%m1qn3%n_iter`	set in config file (1500)
`maxstore`	int	size of available storage	`config%m1qn3%n_iter`	set in config file (1500)

Table 3.2: Variables called by `ropp_1dvar_minropp` routine in ROPP 1DVar

4 ROPP 1dVar results

The performance of the ROPP 1dVar retrieval using `M1QN3` and `minROPP` minimisers has been compared. The stand-alone tools `ropp_1dvar_bangle` and `ropp_1dvar_refrac` have been applied to retrieve temperature, humidity and pressure profiles from the bending angle and refractivity observations included in the ROPP test procedure `IT-1DVAR-03`. This procedure includes data from 8 different radio occultation scenarios for testing.

Tables 4.1-4.6 list the cost function values at each iteration for the different observed profiles processed using `M1QN3` and `minROPP`. Results are listed for the 1dVar retrieval using bending angle and refractivity observations. The maximum relative change in the state vector is also listed for each iteration. This change is computed as a fraction of the background error covariance. Convergence is assumed to occur when the maximum fractional change detected between iterations is less than 0.1 for two successive iterations. The processing time for each 1dVar retrieval is listed for each case.

Figures 4.1-4.8 show plots of the retrieved temperature, humidity and pressure profiles for each of the `IT-1DVAR-03` occultations processed. The difference between results obtained using `M1QN3` and `minROPP` are plotted in green for bending angle observations and blue for refractivity observations. These plots quantify the impact of using the new ROPP-specific minimiser on 1dVar retrievals compared with the `M1QN3` minimiser provided with ROPP v1.0.

4.1 Summary

These tests indicate that the performance of the new `minROPP` is very similar to that of `M1QN3`. Differences in the rate of convergence and cost function values in Tables 4.1-4.6 can be attributed to very small numerical differences ($\approx 10^{-8}$) resulting from rounding errors between the two different codes. The impact of these differences on the cost function and state vector is cumulative between successive iterations. The logic and processing of `M1QN3` and `minROPP` is otherwise identical. Comparison of the CPU time taken to minimise the cost function for each retrieval demonstrates that use of `minROPP` in place of `M1QN3` does not incur any significant losses or gains in processing time. Further optimisation of `minROPP` is however possible by reducing the size of storage available for s and y vectors (`maxstore`) to a smaller value. A value of 20 was sufficient for the tests conducted here.

The retrieved profiles plotted in Figures 4.1-4.8 demonstrate the impact of the numerical differences between `M1QN3` and `minROPP` on the 1dVar output. Maximum differences in temperature of 0.2 K, in specific humidity of 0.05 g/kg and in pressure of 0.4 hPa are well within the quality tolerances required.

The choice of minimiser used in ROPP therefore has minimal impact on the retrieved atmospheric profiles, and it is strongly recommended that users implement the new ROPP-specific minimiser `minROPP` for their applications in the next release of ROPP, with no impact on data quality expected.

4.2 Convergence rates

4.2.1 IT-1DVAR-03 profile 1

`M1QN3`			`minROPP`		
`n_iter`	J	max. relative change state	`n_iter`	J	max. relative change state
1	206.48	-	1	206.48	-
2	120.35	.34257	2	120.35	.34257
3	63.985	.54345	3	63.985	.54345
4	52.375	.17836	4	51.710	.18532
5	44.413	.11091	5	43.107	.12668
6	31.633	.25184	6	32.658	.20357
7	25.640	.12977	7	27.754	.17996
8	19.131	.21257	8	23.469	.91979E-01
9	15.758	.17157	9	19.841	.12733
10	13.882	.60927E-01	10	15.484	.18383
11	12.402	.89095E-01	11	13.517	.14162
			12	12.790	.58348E-01
			13	12.101	.14093E-01
CPU time taken: 0.261 s			CPU time taken: 0.362 s		

Table 4.1: Comparison of cost function values at each iteration applying the ROPP 1dVar to bending angle observations and Profile 1 in the ROPP 1dVar module test IT-1DVAR-03 using the `M1QN3` and `minROPP` minimisers.

`M1QN3`			`minROPP`		
`n_iter`	J	max. relative change state	`n_iter`	J	max. relative change state
1	62.523	-	1	62.523	-
2	51.090	.17426	2	51.090	.17426
3	18.382	.89338	3	18.382	.89338
4	14.214	.19408	4	13.644	.22445
5	8.8994	.38647	5	9.1786	.32950
6	7.7327	.24768	6	8.1835	.24815
7	7.5740	.13512	7	7.6699	.95128E-01
8	7.4954	.50678E-01	8	7.5698	.36934E-01
9	7.4912	.50285E-02			
CPU time taken: 0.193 s			CPU time taken: 0.186 s		

Table 4.2: Comparison of cost function values at each iteration applying the ROPP 1dVar to refractivity observations and Profile 1 in the ROPP 1dVar module test IT-1DVAR-03 using the `M1QN3` and `minROPP` minimisers.

4.2.2 IT-1DVAR-03 profile 2

M1QN3			minROPP		
`n_iter`	J	max. relative change state	`n_iter`	J	max. relative change state
1	143.77	-	1	143.77	-
2	118.05	.12560	2	118.05	.12560
3	52.345	.66877	3	52.345	.66877
4	41.185	.20322	4	39.942	.22035
5	33.272	.16718	5	32.201	.20897
6	25.145	.19800	6	26.144	.13673
7	18.217	.19741	7	20.852	.18598
8	14.157	.20570	8	18.031	.17073
9	12.107	.12276	9	15.077	.10041
10	10.506	.11588	10	12.978	.14324
11	9.2992	.10583	11	11.610	.10368
12	8.6456	.11681	12	10.676	.54124E-01
13	8.4972	.21000E-01	13	9.9476	.52200E-01
14	8.4248	.15080E-01			
CPU time taken: 0.360 s			CPU time taken: 0.364 s		

Table 4.3: Comparison of cost function values at each iteration applying the ROPP 1dVar to bending angle observations and Profile 2 in the ROPP 1dVar module test IT-1DVAR-03 using the `M1QN3` and `minROPP` minimisers.

M1QN3			minROPP		
`n_iter`	J	max. relative change state	`n_iter`	J	max. relative change state
1	50.379	-	1	50.379	-
2	41.130	.17235	2	41.130	.17235
3	13.617	.91738	3	13.617	.91738
4	10.833	.19446	4	10.534	.21998
5	7.8301	.44252	5	7.7738	.43272
6	8.0447	.18259	6	7.6814	.14294
7	7.5978	.10608	7	7.5600	.54936E-01
8	7.5620	.19387E-01	8	7.5558	.76194E-02
9	7.5557	.91877E-02			
CPU time taken: 0.192 s			CPU time taken: 0.185 s		

Table 4.4: Comparison of cost function values at each iteration applying the ROPP 1dVar to refractivity observations and Profile 2 in the ROPP 1dVar module test IT-1DVAR-03 using the `M1QN3` and `minROPP` minimisers.

4.2.3 IT-1DVAR-03 profile 3

`M1QN3`			`minROPP`		
`n_iter`	`J`	max. relative change state	`n_iter`	`J`	max. relative change state
1	83.261	-	1	83.261	-
2	68.185	.12084	2	68.185	.12084
3	28.811	.56841	3	28.811	.56841
4	23.043	.21358	4	22.404	.22315
5	16.775	.37981	5	16.145	.33264
6	13.539	.27744	6	14.202	.34453
7	11.908	.82347E-01	7	12.371	.77892E-01
8	11.050	.12259	8	11.739	.43259E-01
9	10.809	.72700E-01			
10	10.628	.46099E-01			
CPU time taken: 0.251 s			CPU time taken: 0.213 s		

Table 4.5: Comparison of cost function values at each iteration applying the ROPP 1dVar to bending angle observations and Profile 3 in the ROPP 1dVar module test IT-1DVAR-03 using the `M1QN3` and `minROPP` minimisers.

`M1QN3`			`minROPP`		
`n_iter`	`J`	max. relative change state	`n_iter`	`J`	max. relative change state
1	38.233	-	1	38.233	-
2	31.371	.18013	2	31.371	.18013
3	14.611	.59889	3	14.611	.59889
4	11.837	.26804	4	11.515	.29331
5	9.7403	.52943	5	9.9424	.50396
6	9.2436	.11805	6	9.4671	.21192
7	9.1388	.40327E-01	7	9.2140	.69894E-01
8	9.1052	.35843E-01	8	9.1651	.24303E-01
CPU time taken: 0.175 s			CPU time taken: 0.183 s		

Table 4.6: Comparison of cost function values at each iteration applying the ROPP 1dVar to refractivity observations and Profile 3 in the ROPP 1dVar module test IT-1DVAR-03 using the `M1QN3` and `minROPP` minimisers.

4.3 Retrieved profiles

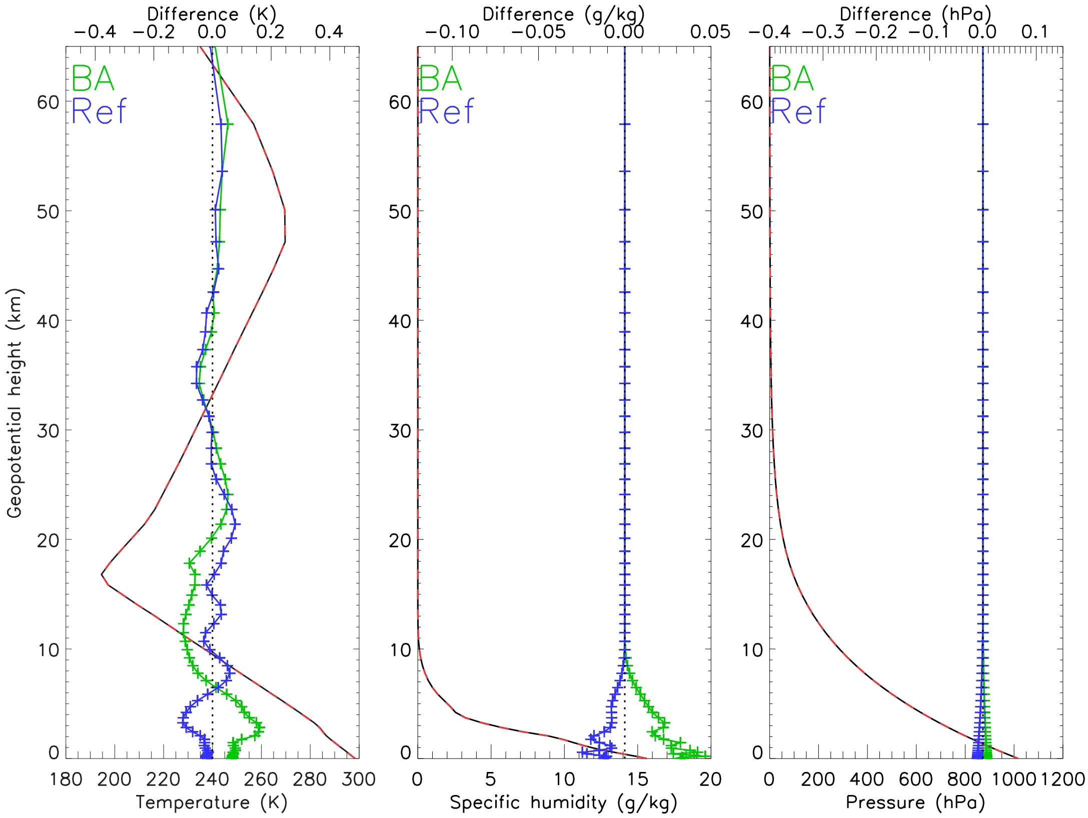

Figure 4.1: Comparison of retrieved profiles of temperature, humidity and pressure for Profile 1 in the ROPP 1dVar module test IT-1DVAR-03 using the `M1QN3` and `minROPP` minimisers. The difference in profiles resulting from the difference of minimiser (`M1QN3-minROPP`) is plotted in green for 1dVar using bending angle observations and in blue for refractivity observation.

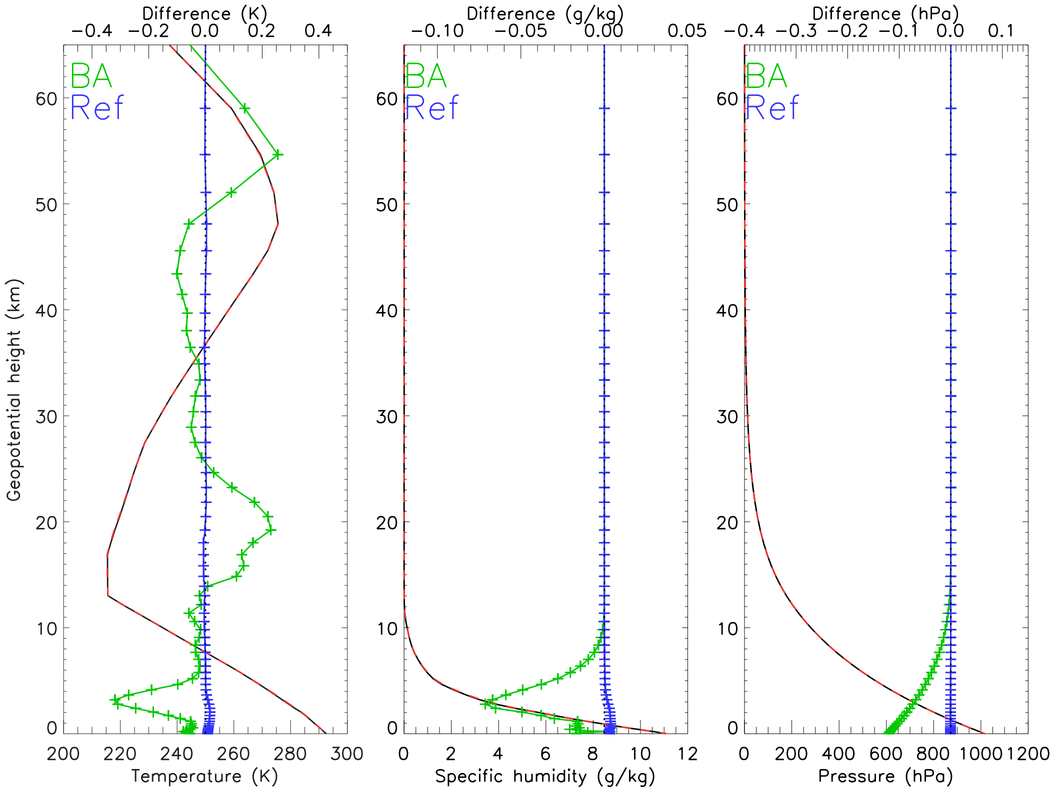

Figure 4.2: As in 4.1 for Profile 2.

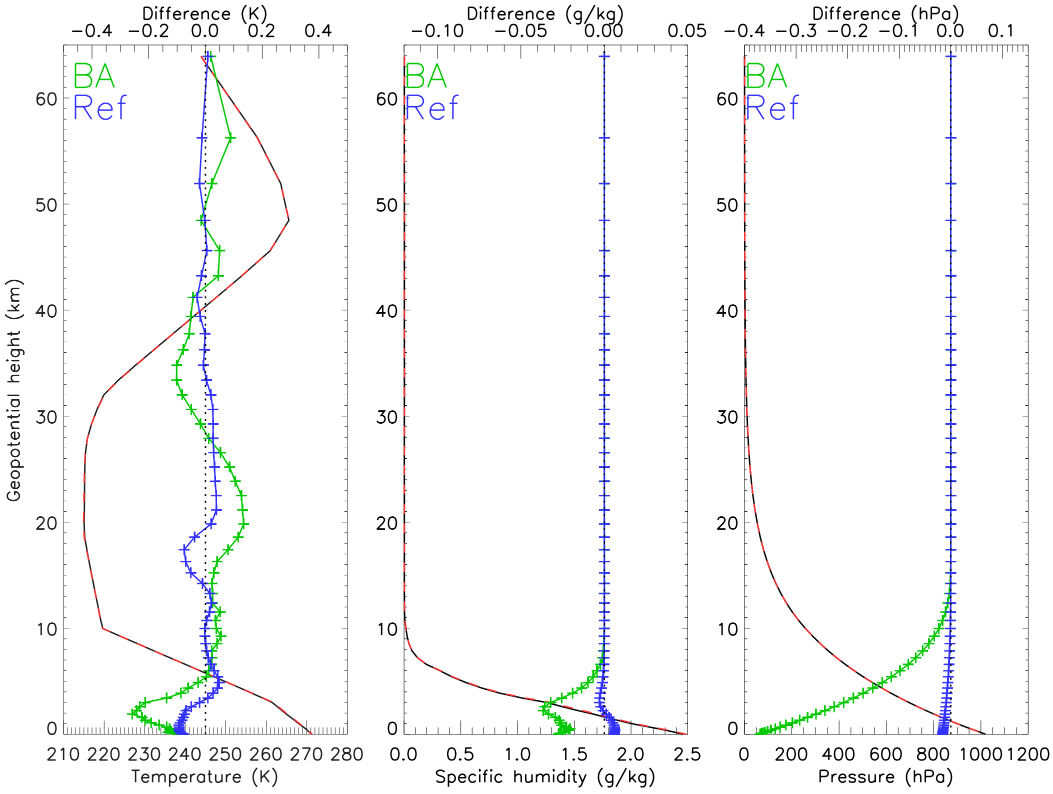

Figure 4.3: As in 4.1 for Profile 3.

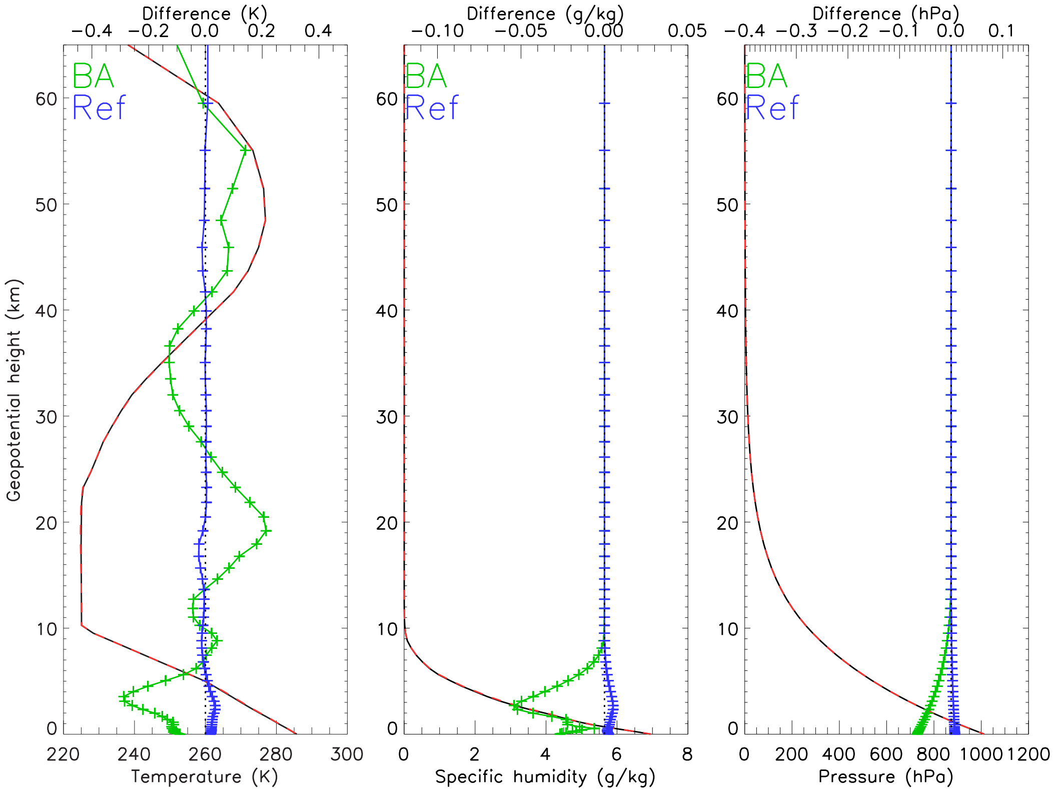

Figure 4.4: As in 4.1 for Profile 4.

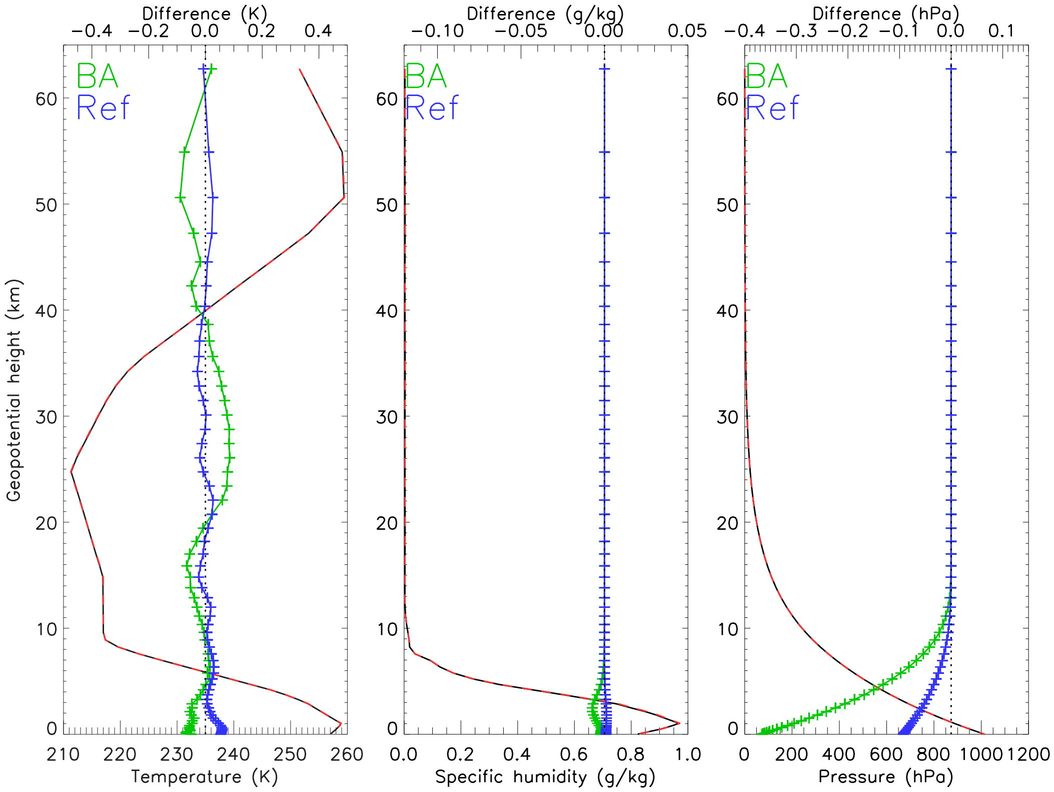

Figure 4.5: As in 4.1 for Profile 5.

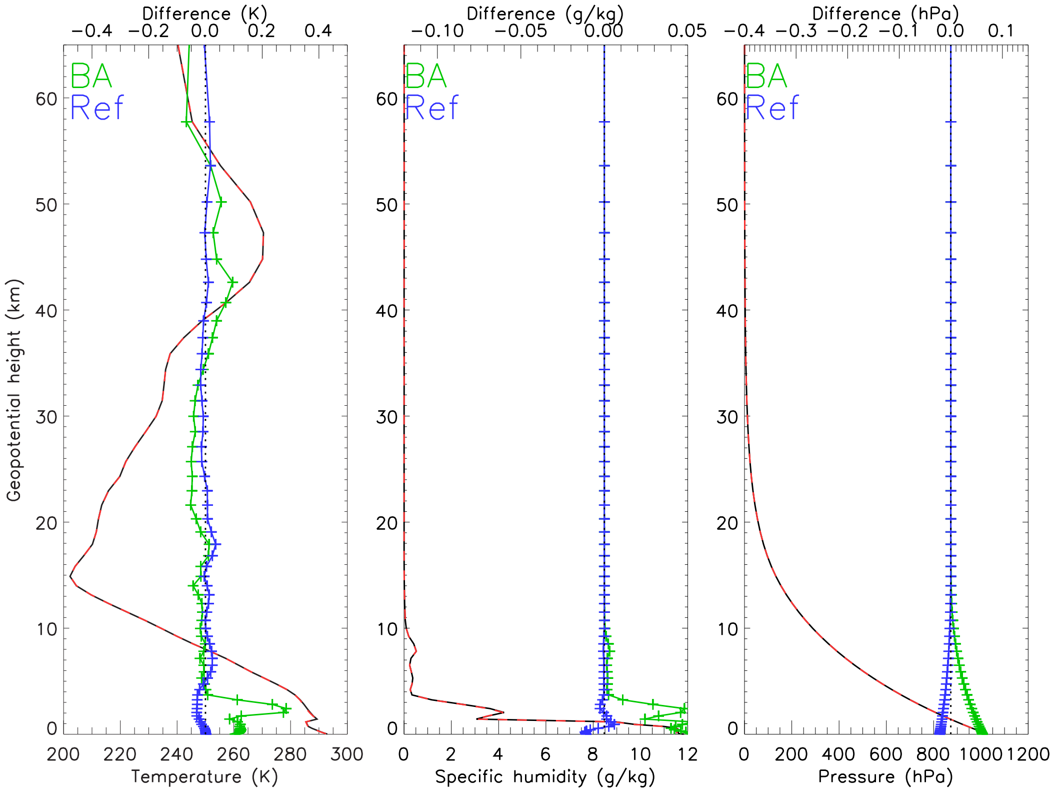

Figure 4.6: As in 4.1 for Profile 6.

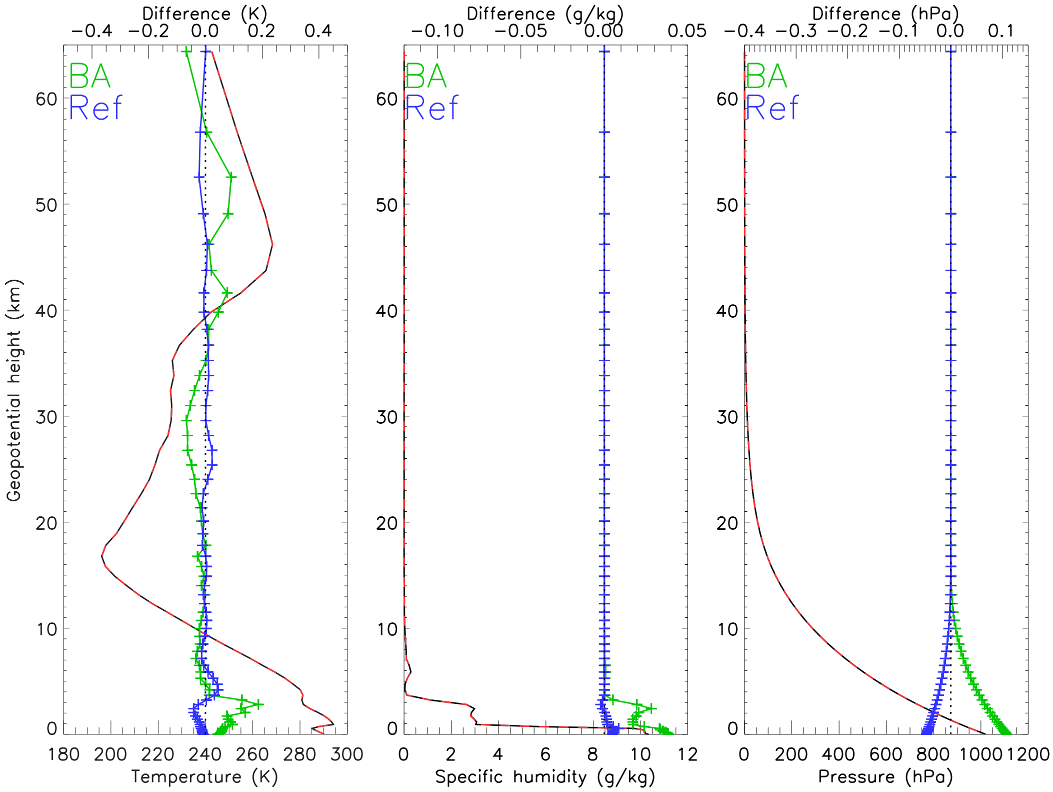

Figure 4.7: As in 4.1 for Profile 7.

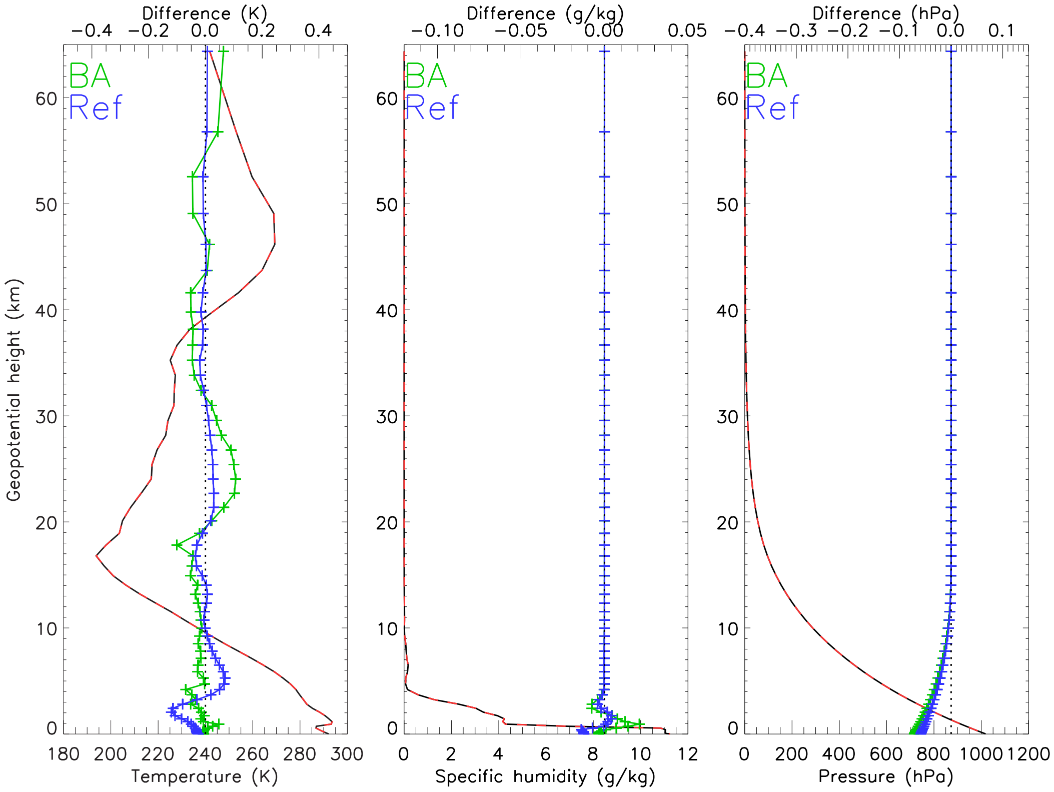

Figure 4.8: As in 4.1 for Profile 8.

Bibliography

[1] Gilbert, J. C. and Lemaréchal, C. M1QN3. A solver of large-scale unconstrained minimization problems, http://www-rocq.inria.fr/estime/modulopt/optimization-routines/m1qn3/m1qn3.html.

[2] Allan, D. (2006) Large-scale minimisation in Meteorology. MSc thesis, University of Bath.

[3] Gilbert, J. C. and Lemaréchal, C. (1989) Some numerical experiments with variable storage quasi-Newton algorithms. *Mathematical Programming* **45** 407-435

[4] Nocedal, J. (1980) Updating quasi-Newton matrices with limited storage. *Mathematics of Computation* **35 773-782**

GRAS SAF Reports

SAF/GRAS/METO/REP/GSR/001	Mono-dimensional thinning for GPS Radio Occulation
SAF/GRAS/METO/REP/GSR/002	Geodesy calculations in ROPP
SAF/GRAS/METO/REP/GSR/003	ROPP minimiser - minROPP
SAF/GRAS/METO/REP/GSR/004	Error function calculation in ROPP
SAF/GRAS/METO/REP/GSR/005	Refractivity calculations in ROPP
SAF/GRAS/METO/REP/GSR/006	Levenberg-Marquardt minimisation in ROPP
SAF/GRAS/METO/REP/GSR/007	Abel integral calculations in ROPP
SAF/GRAS/METO/REP/GSR/008	ROPP thinner algorithm
SAF/GRAS/METO/REP/GSR/009	Refractivity coefficients used in the assimilation of GPS radio occultation measurements
SAF/GRAS/METO/REP/GSR/010	Latitudinal Binning and Area-Weighted Averaging of Irregularly Distributed Radio Occultation Data
SAF/GRAS/METO/REP/GSR/011	ROPP 1dVar validation

GRAS SAF Reports are accessible via the GRAS SAF website http://www.grassaf.org.