RESPONSE TO REFEREE COMMENTS for the ARTICLE

Quasi-Newton Methods for Atmospheric Chemistry Simulations: Implementation in UKCA UM vn10.8

Emre Esenturk[1,2], Nathan Luke Abraham[2,3], Scott Archer-Nicholls[2], Christina Mitsakou[2,b], Paul Griffiths[2,3], Alex Archibald[2,3], John Pyle[2,3]

We thank the anonymous referees for their valuable comments and time.

Referee 1

Overall, the manuscript is written very carefully and is easy to read. I can find hardly any typos or awkward sentences. Please fix the Table 2 title, FN should be NR I believe?
What I cannot understand is how the quasi-Newtonian method (QN), which provides only an approximation of the true Jacobian use in the standard Newton-Raphson (NR) method, is able to find an answer within the radius of convergence and thus avoid the NR method 'wandering around the wasteland of bad solutions'. I would expect this ability from Markov Chain Monte Carlo methods, but not QN. Do the authors have any explanation for this?

The implemented the QN method which is, as the referee points out, only an approximation manages to stay in the radius of convergence partly because it is fed and backed by the NR step. One could write the code in a way that QN method alone takes the process after the initial initialisation. However, in general this will not work since, in a stiff environment like the atmosphere, after a few consecutive QN steps the approximate Jacobian will fall far from the true value and hence the program will enter the "wasteland" and the solutions will go astray. In that sense this version of the QN method utilizes the NR step instead of completely avoiding it.

We briefly mention this in the introduction and give some more information in Section 2.4 (two paragraphs before Equation 7). We added the following text to Section 1 (the second paragraph after Figure 1) in order clarify this point:

"A key point of the implementation is that the additional internal QN iterations do not replace the NR iterations completely. Rather each QN iteration works in and is fed by the current NR iteration."

p.6/l.10 The deep-seated drawback of implicit schemes is that they are inherently only first-order in time. (At least all the ones I have played with). People are developing higher order tracer transport schemes (as in Lauritzen's papers), but with first-order chemical solvers, we are left with first-order errors. In fact, the authors analysis of errors in QN vs. RN later in the paper might do

better to compare these with halving the time step. I think the relative errors shown here are all trivial.

Since the method is quite generic as far as the ODE scheme is concerned, it would also be possible to test and compare it with numerical schemes other than backward Euler. The current numerical scheme is partly hard wired. It can be an interesting future problem to do this comparison. We thank the referee for the new references. It is now included in the references.

p.6/l.22 (This is an aside) I am very surprised that UKCA still uses the old tricks of putting emissions and deposition into the kinetics. This is highly unstable since it forces these terms to be absorbed into a single grid cell for the full chemical time step. It becomes totally unstable with increasing vertical (or horizontal) resolution. The boundary layer is mixed on time scales of 30-90 min, and high dep velocities will empty a 50 m layer in the chemical time step. It is good that UKCA works with this coupling, but it may fail depending on intensity of local emissions or deposition.

While the formulation discussed here is to include emissions and deposition in with the kinetics, in fact UKCA treats emissions separately, and these are added during the boundary-layer mixing step that occurs every dynamical timestep. Dry and wet deposition are still treated as described, although dry-deposition acts throughout the whole of the boundary layer. Wet deposition follows Giannakopoulos et al 1999. We have amended the text to clarify the treatment of emissions with the following:

"In the current implementation, emissions are treated separately during the boundary-layer mixing step, and dry deposition occurs throughout the boundary layer."

Giannakopoulos, C., M. P. Chipperfield, K. S. Law, and J. A. Pyle (1999), Validation and intercomparison of wet and dry deposition schemes using 210Pb in a global three-dimensional off-line chemical transport model, J. Geophys. Res., 104(D19), 23761–23784, doi:10.1029/1999JD900392.

*p.10. This is a very nice derivation. In our model, we stop re-evaluating and solving the full Jacobian when our relative errors drop below some threshold (0.1 or 0.03). Within the radius of convergence, just iterating on the right-hand side vector (your c-sup-k) C2 GMDD converges almost as swiftly and much more cheaply. I had not seen the factor (1 – asup-k) before and will be interested to try it out. You discuss correctly about the lower operation count of QN vs NR, but it might be useful to note here the simple numerics that solving J*delta-c = -f(c) requires nˆ3 /3 operations (inverting J requires nˆ3) but that resolving this as in eqn 10 requires only nˆ2 ops.*

That how much the *re-solving operation* saves, as opposed to regular solving operation, is a nice point to note. We thank the referee for pointing out/highlighting this to us. We added the following text to the end of Section 2

"In practical terms, this means that ~N^3 numerical operations that are normally needed to solve a linear system is now reduced to ~N^2 operations which gives substantial savings within the routine. An example of the implementation of these changes is given in the pseudo-code provided in Appendix A."

p.12. Can you explain simply about the CPU time vs wall clock time. When we run, the CPU time is usually 6 times the wall clock time (effectively we are using 6 out of the 8 cpus on the board).

The box-model was run with a single core, hence the wallclock was approximately equal to the CPU time. The text has been modified to say the box-model was run on a single core:

"All UKCA_BOX experiments were run on a single processor core."

"Table 2. Wallclock times for running 1000 calls for the NR iterations and QN iteration within the UKCA_BOX model run on a single processor core."

p.14/l.12. I would not have thought that 'stiffness' applied to any given iteration, but rather it was a property of the Jacobian (and the system as a whole).

Here we meant that the solution was more challenging to find on some timesteps than others, generally at the transition between light and dark. The text has been modified accordingly.

"The first timestep is the most difficult to solve, as the initial chemical concentrations are typically far from a steady state having been taken from monthly average values from model cells."

p.16. Regarding the analysis of Fig 4, I would not exaggerate the 1st hour since the initial conditions (some mean values) are horribly off from the correct answer and this almost never happens in a continuously running model. The changes over twilight are a globally common problem and very important to compute efficiently.

We would agree that the first timesteps are not a particularly realistic scenario, the text has been modified to focus more on the twilight steps and less on the first timestep by removing a few statements discussing analysis of the first timestep. However, we would note that the first timestep takes the greatest number of iterations to solve (12), and while rare there are many gridcells which take 12 or more to solve in the 3D model. So while the exact chemical mix of the first timestep may not be representative, the computational challenge is, and showing whether the solver can handle the first timestep or not is a good test of its robustness.

p.18. Table 5 is a bit hard to figure out. I see that QN2-3 decreases chemistry, but its increase in convection is artificial since the total time decreased. Is there and easier way to do this? Also, it would be nice to see the timings for all the components.

We have updated Table 5 giving the timings in seconds for the simulations performed. It is difficult to get timings for all components, as these are provided subroutine name rather than by subcomponent from the Dr Hook timing routines. We have added timings for radiation and dynamics to the table however, as well as the total time for the simulation.

The new table is as follows:

Average wallclock time in seconds (± 2 standard error) across all processors used for various UM components comparing the CNTL and QN2-3 methods. All are from 1-year simulations performed on a Cray XC40.

Chemistry	***StratTrop***		***StratTrop+GLOMAP***		***StratTrop***	
Cores	***432***		***432***		***216***	
Simulation	***CNTL***	***QN2-3***	***CNTL***	***QN2-3***	***CNTL***	***QN2-3***
Dynamics	*12123 ± 22*	*12099 ± 23*	*15117 ± 28*	*15297 ± 27*	*18881 ± 27*	*18743 ± 30*
Chemistry	*4228 ± 26*	*3678 ± 16*	*4725 ± 28*	*4123 ± 19*	*9102 ± 96*	*7875 ± 75*
Diagnostics	*2951 ± 1*	*2979 ± 1*	*3628 ± 1*	*3641 ± 1*	*3098 ± 1*	*3108 ± 1*
Photolysis	*3038 ± 7*	*3038 ± 7*	*3041 ± 7*	*3030 ± 7*	*6082 ± 43*	*6084 ± 43*
Convection	*1833 ± 51*	*1828 ± 51*	*2367 ± 62*	*2366 ± 62*	*3648 ± 148*	*3637 ± 148*
Radiation	*1184 ± 10*	*1184 ± 10*	*1140 ± 10*	*1136 ± 9*	*2487 ± 34*	*2485 ± 34*
UM Total	*48871 ± 0*	*47730 ± 0*	*71900 ± 0*	*70596 ± 0*	*82561 ± 0*	*79600 ± 0*
Chemistry Speed-up (%)	*13.00*		*12.74*		*13.48*	
UM Speed-up (%)	*2.33*		*1.81*		*3.59*	

p.19/l1-14. I am surprised that you allow 40 iterations with NR. Since convergence is quadratic, relative errors for the last 4 iterations go as: e-1, e-2, e-4, e-8. Thus the problem is to get within the radius of convergence (say, e-1). If our NR solver does not find a solution within 7 iterations, it is time to try a new starting guess or cut the time step as you note. It would be interesting to single out and diagnose what is going wrong when a few grid cells hold up all. We found that happened when our convection lofted very-high isoprene values to the upper trop. So the fundamental problem is finding a better starting guess.

The choice of 50 iterations was made for the original formulation of the chemical solver used here, and described in Wild & Prather (2000), and has not been changed since. It should be noted that a QN step is always paired with a full NR step, and so convergence is never determined on a QN step alone.

Wild, O., and M. J. Prather (2000), Excitation of the primary tropospheric chemical mode in a global three- dimensional model, *J. Geophys. Res.*, 105(D20), 24647–24660, doi:10.1029/2000JD900399.

p.21. Very nice.
p.25 Table 7 is impressive, and I suspect dwarfed by the delta-t errors.

We thank the reviewer for his/her positive and useful feedback. It not only helped us to improve and polish the text but also gave us further thoughts on possible ways of improving our current code for the future versions.

Referee 2

You state that 'each QN call costing 27% of a call to the full NR routine.'. I would recommend this is made more specific with regards to overall cost to the entire box-model simulation. At the moment the cost is presented as an individual function call. Whilst an interesting statistic, and I appreciate different solvers will use different approaches to Jacobian iterations, this would be a valuable take-home message. As the authors note in other areas of the paper, some large scale models have associated costs of ~70% total run-time from the chemical routine.

The box model results show a reduction in the mean number of full NR equivalent iterations from 4.36 to 3.89, or 12% less, which resulted in a reduction in total runtime of less than 1%. The global model results show 26% reduction in the mean number of full NR equivalent iterations from 6.06 to 4.48 for 432-core StratTrop, but resulted in a 2.33% reduction in run-time. While the solver is a key part of the both the box and global models, there are many other steps such as initialisation, I/O etc., that all take time. In the 3D model, the longer simulations mean that some things are less important (initialisation for example), but the very short times for the box model make it especially sensitive to initialisation, I/O, and load on the system. The box model simulations themselves only take around 2 seconds each, and so the box model is more useful for developing a functional understanding of how the mixed-QN methods work, and as a test-bed we can use to optimise our developments, rather than as a simplified example which can be scaled to the global model, and the discussion in the paper was written to reflect this. In the box model, it can move on as soon as the chemistry in a single grid cell has converged to a stable solution, whereas in the 3D model each core needs to wait for all the other cores to finish before moving onto the next timestep, i.e the model is only as fast as its slowest member. There are important benefits from keeping processors more in-line with each other and reducing the occurrence of timestep-halving which are simply not represented in the wallclock time of the box-model. For these reasons, we consider presentation of the runtime of the box model to be misleading; the main purpose of this paper is to show how these developments improve model performance in the 3D model, not the box model.

In the abstract you also state that 'The blended QN method also improves the robustness of the solver, reducing the number of grid cells which fail to converge after 50 iterations. ' Can you put this in wider context? What are the typical number of grid cells that would otherwise fail to converge after 50 iterations? Even if the efficiency gain is small, this could be an important reason for adopting the method. I appreciate space is tight in the abstract, but do make sure this is covered elsewhere.

The information in Table 6 lists the number of times a halving step is performed. The QN method reduces the need for this by 40%. We have added this information to the abstract. In the configuration of UKCA used here, the solver is called over each vertical level in the domain, and we are unable to tell how many grid-cells within that layer failed to converge. The sentence in the abstract now reads:

The blended QN method also improves the robustness of the solver, reducing the number of grid cells which fail to converge after 50 iterations by 40%.

On the general cost saving point above, do the authors feel a 3% reduction in cost is significant for the UKCA? Is this within cost variability for other implicit solvers available? This might be an interesting comparison, even within the box-model simulations.

In the context of the 11,200 model years of CMIP6 simulations currently underway using the UKESM1 Earth System Model (which includes this development), a 3% saving is equivalent to saving the time to simulate over 300 model years. This is roughly 100 real days supercomputer time. Discussion of this point has been added to the conclusion:

"Overall, we see a reduction in total computational costs of the whole UM-UKCA model of approximately 3%, corresponding to a reduction of approximately 15% in the chemistry routines. Whilst this may not seem like a big reduction, it is significant given the high costs associated with the rest of the coupled UM-UKCA model. In practice, a 3% reduction of costs for a large study involving 10,000 model years corresponds to 300 model years saved, roughly 100 real days of supercomputer-time with the current setup"

We have not tested this method with other solvers, although it should be noted that the savings within the solver were 13%. The box model used here is designed as a tool to aid in UKCA model development, and so it is unable to run other solvers not included in UKCA.

Most Jacobian matrices in atmospheric mechanisms are sparse. Is there any perceived scaling issues with sparsity with your method?

We have not tested this method recently with different chemistry schemes. We have performed tests at previous UM versions using the stratospheric chemistry of Morgenstern et al (2009), the tropospheric chemistry of O'Connor et al (2014), and the StratTrop chemistry that is produced

when these two schemes are combined (as has been used in this study). These showed that the changes in run-times were essentially linear between the three schemes.

Section points: Section 1, page 3 lines 5 onwards. Here you discuss 'construction' of a Jacobian. Defining the Jacobian of a gas phase solver is relatively easy and, whilst pedantic, I'm unsure if the authors are referring to explicit calculation of a Jacobian during solver iterations or some approximation via finite difference. Please clarify …

For constructing the Jacobian we use explicit and exact forms *not* the finite difference approximation which would increase the costs. Where we do the approximation is in the pseudo iteration step and it is an approximation in a different sense.

Section 2.3. I'm interested in the strategy in the existing solver for testing convergence before reducing the time-step and re-calculating. Presumably this is a heterogeneous issue on any given domain, but how are 50 iterations chosen as a point to reduce the time-step and wouldn't an adaptive method work?

As mentioned in the response to Reviewer 1, the choice of 50 iterations was made in the original formulation of the solver prior to it being adopted by UKCA, and Table 6 shows that this halving step occurs in less than 0.3% of solver calls. Convergence is tested for any point in the vertical layer in the processor domain that has been passed to the solver, and if any gridcell within the layer is not converged the solver continues to iterate. There are a number of possible solutions to reducing the computational effort used here, and these are being actively investigated. While an adaptive method to reduce the timestep may be beneficial, it could also add more time during the checking step.

Section 3.2.1 page 18. The authors make an interesting comment on potential speedups for 'spatial systems modelled by partial differential equations'. This includes reaction-diffusion problems, or processes in particulates. However it is not clear how generally applicable the method could be. Is it possible the method could ever lead to a decrease in performance?

Generally the method will lead to an increase in performance unless all "future" Jacobians are constructed solely by the approximate scheme in *stiff* systems. This is the reason why, in our stiff problem, we refresh (i.e., exactly compute) the Jacobian in selected iterations. If the system is mildly stiff the method will work more effectively and one may even set a_k=0 in Equation 10 which amounts to using the same Jacobian for multiple times which would save even more time. However one needs to be careful as overusing the same Jacobian can cause substantial deviation from the true Jacobian and may take the iterates far off the solution.

Section 3.2.2, page 22 line 7: 'If the 3D model predictions for the two species which are on the opposite sides of the lifetime spectrum are very close, then it is very likely that physical values for all other species which have intermediate lifetimes will also be close.' Is it not relatively easy to demonstrate the range of propagation errors in all species? I would suggest a demonstration of

this is included rather than a qualitative statement on potential. If this is not straight forward, please state why

Following the suggestion of the referee we computed the normalised mean absolute difference (NMAD), normalised root mean square difference and normalised mean bias (NMB) values for all species after 10 (model) years and 20 years, and have included these in below and in the supplement and added the following sentences right before Table 7 and after Table 7 which read:

A complete table showing NMAD, NRMSD and NMB for all species is provided in the supplement.

Similar conclusions can be drawn for the other species as shown in the supplement

c) NMAD, NRMSD and NMB Values for all Species

Species	**NMAD (*E-3) After 120 months**	**NMAD (*E-3) After 240 months**	**NRMSD (*E-3) After 120 months**	**NRMSD (*E-3) After 240 months**	**NMB (*E-3) After 120 months**	**NMB (*E-3) After 240 months**
O3	0.0011	0.00088	0.0010	0.00090	0.00098	0.00074
NO	0.0441	0.0340	0.0066	0.0077	-0.0032	-0.0020
NO3	0.0027	0.0040	0.0074	0.0170	-0.0011	-0.0021
N2O5	0.0045	0.0027	0.0052	0.0074	-0.0037	-0.0016
HO2NO2	0.0053	0.0059	0.0079	0.0086	-0.0013	-0.0015
HNO3	0.0066	0.0058	0.0074	0.0116	-0.0046	-0.0025
H2O2	0.0150	0.0172	0.0332	0.0423	-0.0135	-0.0158
CH4	0.0003	0.0003	0.0004	0.0004	0.0003	0.0003
CO	0.0024	0.0025	0.0019	0.0020	0.0017	0.0017
HCHO	0.0089	0.0101	0.0276	0.0343	-0.0033	-0.0050
CH3OOH	0.0089	0.0105	0.0180	0.0220	-0.0049	-0.0064
HONO	0.0206	0.0204	0.1723	0.2035	0.0017	0.0040
C2H6	0.0118	0.0137	0.0098	0.0129	0.0118	0.0136
C2H5OOH	0.0156	0.1939	0.0186	0.0279	0.0085	0.0111
CH3CHO	0.0098	0.0012	0.0010	0.0013	0.0056	0.0067

PAN	0.0115	0.0153	0.0319	0.0530	-0.0011	-0.0006
C3H8	0.0092	0.0127	0.0071	0.0140	0.0089	0.0123
n-C3H7OOH	0.0140	0.0225	0.0189	0.0412	0.0103	0.0180
i-C3H7OOH	0.0152	0.0237	0.0198	0.0412	0.0103	0.0180
C2H5CHO	0.0093	0.0134	0.0408	0.0373	0.0072	0.0101
C2H6CO	0.00933	0.01099	0.00783	0.00959	0.00926	0.01086
CH3COCH2OOH	0.01394	0.01660	0.01950	0.02661	0.00301	0.00494
PPAN	0.01001	0.01193	0.03702	0.06292	-0.00931	-0.01031
CH3ONO2	0.01053	0.01355	0.01481	0.01974	-0.00932	-0.01286
C5H8	0.02416	0.02421	0.02180	0.02464	0.01083	0.01182
ISOOH	0.02798	0.02933	0.03170	0.03713	0.01167	0.01110
ISON	0.03209	0.04159	0.06750	0.10933	0.00375	0.01014
MACR	0.02345	0.02557	0.02783	0.03507	0.01884	0.01998
MACROOH	0.04468	0.05189	0.05149	0.07872	0.02563	0.03051
MPAN	0.03073	0.03666	0.04900	0.06991	-0.00697	-0.00842
HACET	0.02630	0.03193	0.03694	0.05458	-0.00538	-0.00461
MGLY	0.14927	0.18045	0.41070	0.63219	-0.14196	-0.17186
NALD	0.04737	0.06276	0.10608	0.17306	0.02292	0.04034
HCOOH	0.01836	0.02537	0.02768	0.04264	-0.00854	-0.00811
CH3CO3H	0.02581	0.03148	0.03700	0.05843	0.01038	0.01493
CH3CO2H	0.02374	0.02903	0.03652	0.05757	-0.00118	0.00231
MVKOOH	0.02997	0.03284	0.05591	0.06614	0.00713	0.00526
Cl	0.00069	0.00035	0.00090	0.00036	-0.00069	-0.00020
ClO	0.00245	0.00247	0.00738	0.00731	0.00015	0.00064
Cl2O2	0.09651	0.05123	0.28837	0.05308	0.00543	0.02153
OClO	0.01277	0.01082	0.01654	0.01336	0.00894	0.00849
Br	0.00051	0.00056	0.00048	0.00065	-0.00029	-0.00023
BrCl	0.01104	0.00941	0.01503	0.01262	0.00771	0.00731
BrONO2	0.00471	0.00439	0.00757	0.00754	-0.00234	-0.00186
N2O	0.00004	0.00003	0.00008	0.00005	0.00004	0.00003

HOCl	0.00639	0.00587	0.01665	0.01657	0.00087	0.00045
HBr	0.008348	0.00908	0.01627	0.015961	-0.00754	-0.0083
HOBr	0.006061	0.00595	0.011883	0.012055	0.00258	0.00168
ClONO2	0.001965	0.00217	0.00272	0.0031523	-8E-05	0.00043
CFCl3	4.35E-05	3.9E-05	8.4E-05	7.53E-05	4.3E-05	3.8E-05
CF2Cl2	3.54E-05	2.7E-05	7E-05	5.03E-05	3.3E-05	2.5E-05
CH3Br	0.00044	0.00047	0.00052	0.00055	0.00044	0.00047
N	0.000798	0.00403	0.0006	0.004242	-0.00077	-0.0027
O(3P)	2.66E-05	1.3E-05	4.6E-05	3.19E-05	-1.9E-05	-5E-06
ORGNIT	0.056804	0.06181	0.08377	0.107436	0.03981	0.04092
H	8.77E-06	5.4E-06	2.7E-05	1.99E-05	8.8E-06	2.2E-06
OH	3.64E-05	3.7E-05	3.9E-05	3.6E-05	1.2E-05	3E-06
HO2	0.001041	0.00108	0.00109	0.001262	0.0003	0.00023
CH3OO	0.00291	0.00492	0.00214	0.003857	0.0025	0.00375
C2H5OO	0.056128	0.0569	0.16649	0.16476	0.04521	0.04526
CH3CO3	0.051005	0.05548	0.21106	0.274743	-0.02183	-0.0224
n-C3H7OO	0.076912	0.08898	0.2034	0.272403	0.07119	0.08267
i-C3H7OO	0.070862	0.08225	0.19265	0.258701	0.06177	0.07211
C2H5CO3	0.032232	0.04066	0.11543	0.191884	-0.01635	-0.0173
CH3COCH2OO	0.043401	0.04446	0.11144	0.119976	0.01727	0.0185
CH3OH	0.005674	0.00593	0.007	0.009182	-0.00167	-0.003
HCl	0.000445	0.00049	0.00135	0.001333	-8.3E-05	-0.0001
BrO	0.001422	0.00155	0.00471	0.004844	-0.00062	-0.0006
NO2	0.004671	0.00346	0.00718	0.012697	-0.00346	-0.002
O(1D)	2.69E-05	1.8E-05	3.2E-05	2.43E-05	1.2E-05	1.1E-05

Section 4 Line 11: 'We also demonstrated that, the suggested method..'. Please remove the comma You also state that 'The differences in chemical concentrations between the control run and that using the blended QN method are negligible for longer lived species, such as ozone, ' Please quantify 'negligible'.

We removed the comma and modified the text in the abstract accordingly

blended QN method are of order (10^{-7}) for longer lived species, such as ozone, and below the threshold for solver convergence (10^{-4}) almost everywhere for shorter lived species such as the hydroxyl radical

Section 4 Code Availability. Perhaps I have misunderstood licensing issues here, but this is a slightly disappointing end. Is there a reason why at least an example chemical box-model with the QN method could not be supplied? Even if this was hard-wired, without using a package such as KPP, it fits within the clear procedures and ethos now pervading all GMD papers. I'm not sure if I could reproduce your results and check the potential exciting co-benefit for other models. This could be a simple oversight, but I would suggest the authors check with the paper and editor on providing a minimum statement on the availability of this component. Will the optional QN methods be part of a new release? Maybe I have missed this within the main body of text, or it is implicit with the paper. If this is the subject on on-going work for which the group wishes to retain IP, which is absolutely fine, then I would simply state we could all wait for some exciting follow up studies. If there is a perceived issue with general applicability, this should also be stated.

We produced a pseudocode which is included in the appendix of the paper.

! **Pseudo Code for Solving the Equation *F(c)=0***
! Inside the new chemistry step: determine the concentrations for the next step...

…
…
err = 10^{-4}
…

Update tendencies ($\boldsymbol{f}(\boldsymbol{c}_*)$) at the time of the current chemistry step (t_*)

Make an initial guess for the algebraic system as an input to the iterative solver
$\boldsymbol{c} = \boldsymbol{c}_* + \boldsymbol{f}(\boldsymbol{c}_*)\, del\_t$

! Main NR Iteration loop starts

! Iteration counter: k, maximum iteration counter: max_iter

Do k=1,max_iter

> ! Update the ***F*** vector and store it
> $\boldsymbol{F} = \frac{\boldsymbol{c}-\boldsymbol{c}_*}{del\_t} - \boldsymbol{f}$
> ***Fold =F***
>
> ! Jacobian construction and linear system solving
> Compute exact Jacobian $\mathbf{J}(\boldsymbol{c})$ of the ***F*** vector()
>
> Solve for the new increment $\boldsymbol{del\_c}$ in the equation
> $\mathbf{J}(\boldsymbol{del\_c}) = \boldsymbol{F}$

err_c = $\frac{\text{maxval}(abs(\boldsymbol{delt\_c}))}{maxval(abs(\boldsymbol{c}))}$

! Updating the ***c*** values
Perform treatments for troublesome convergence (e.g. β dampening factor) or
Filtering of possible negative values in components of
$\boldsymbol{c} = \boldsymbol{c} + \beta\, \boldsymbol{del\_c}$

! Test and decide if QN step will be taken
! This can be done on iterations 2 ≤ k ≤ 50, and recommended on steps 2 & 3
! This step will not be done if the ***c*** vector converged and the routine is about to exit

If (err_c ≥ *err* .AND. *choice_qn*) Then

Update the tendencies

Update the ***F*** vector
$\boldsymbol{F} = \frac{\boldsymbol{c}-\boldsymbol{c}_*}{\Delta t} - \boldsymbol{f}$
delF=F-Fold

! **QN approximation below** …

Compute the Jacobian modification factor
$a = \frac{DOT\_product(\boldsymbol{delF},\boldsymbol{F})}{DOT\_product(\boldsymbol{del\_c},\boldsymbol{del\_c})}$

Re-solve for the newer increment ***del_c***
$\mathbf{J}\,(\boldsymbol{del\_c}) = (1-a)\boldsymbol{F}$

Update ***c*** values
$\boldsymbol{c} = \boldsymbol{c} + \beta\, \boldsymbol{del\_c}$

End If

k=k+1

End Do

We thank the referee for his/her valuable comments and time.

Referee 3

The manuscript by Esenturk et al. describes application of a new hybrid numerical algorithm for solving the ODEs governing atmospheric chemistry simulations. The is typically a significant computational bottleneck of such simulations, although here the authors focus on chemistry-climate models whereas the issue is even more pronounced for chemical-transport models. Thus, the topic is generally of interest and suitable for GMD. The methods proposed here are not

ground-breaking, but contain an incremental idea about replacing some Newton-Raphson steps with those using approximate Jacobians (i.e. quasi-Newton). The results are similarly incremental, resulting in speed ups of the overall model by 2-3%, although tests of box-model simulations show up to ~30% improvement. The authors are diligent about exploring different aspects of this problem and considering many ways of evaluating their results, which seem to be fairly sound. I just wish. . . as the authors must. . . that the paper went further in advancing this field. The modest improvements in performance aren't likely to motivate current researches to make a switch to adapt these methods. But perhaps this is a first step in the right direction, and this type of mixed QN/NR method will have improved performance over time. I've provided detailed comments below, which essentially amount to minor revisions.

We would like to thank anonymous Referee #3 for their time in reviewing our manuscript. We are pleased that the reviewer has found our work diligent. We set out with the ambition of increasing the efficiency of our numerical integration scheme and we are pleased with our results. For a very small increase in the number of lines of code we have achieved what we feel is a significant reduction in computational time, as seen in the UKCA_BOX modelling results. Integrating the chemical reactions remains slow but we feel that this is a forward step. Clearly we would have liked a giant leap forward but often this is not the case in science.

1.28: on the –> for the

1.30: can be more specific here? how many fewer grid cells?

As listed in Table 6, the QN method reduces this non-convergence by 40%, although the number of solver-calls where convergence is not achieved within 50 iterations is less than 0.3%.

1.31: Can be more quantitative here as well?

As we also replied to Referee 2, the text has been amended to:

"The blended QN method also improves the robustness of the solver, reducing the number of grid cells which fail to converge after 50 iterations by 40%."

2.26: I object to the statement that "explicit methods are quick" which isn't necessarily true when solving a stiff system and an explicit method is forced to take times steps that are so small that the total integration cost is larger than implicit methods. But, the authors could modify there statement to be correct by changing the scope to any single internal integration step, which indeed are usually quicker for explicit methods than implicit (but still not always).

We thank the referee for noting this point. We are well aware that explicit methods are not always quick. What was meant there was that explicit methods are among the handy choices to give a first go when solving non stiff systems. However, to avoid misunderstanding we modified the text to clarify the point and make it clear that explicit methods are generally quick over single time integration step. But for stiff systems they require very small time steps (to preserve stability) and

become disadvantageous. Hence, for stiff systems, despite possibly taking longer on single step, implicit methods are more advantageous for overall efficiency.

3.11: Is 2004 really a "recent year"? Or are there more recent applications to cite here?

We thank the reviewer for pointing this out. There are many more recent studies and we propose to add a further reference and modify the text to reflect this.

"To overcome the high costs, methods that avoid or reduce Jacobian construction have gained popularity in recent years (Brown and Saad, 1990; Chan and Jackson, 1984; Knoll and Keyes, 2004 and citing literature e.g. Viallet et al., 2016)"

Viallet, Maxime, Tom Goffrey, Isabelle Baraffe, Doris Folini, Chris Geroux, M. V. Popov, Jane Pratt, and Rolf Walder. "A Jacobian-free Newton-Krylov method for time-implicit multidimensional hydrodynamics-Physics-based preconditioning for sound waves and thermal diffusion." *Astronomy & Astrophysics* 586 (2016): A153.

6.15: This approach is commonly implemented in numerous models that could be cited, for example any that uses KPP-generated solvers with sparse Jacobian options enabled.

We thank the referee for the comment. We added the following references to KPP.

Damian, V., Sandu, A., Damian, M., Potra, F., & Carmichael, G. R. (2002). The kinetic preprocessor KPP - a software environment for solving chemical kinetics. *Computers and Chemical Engineering*, *26*, 1567–1579.

Sandu A., Verwer J. G., Blom J. G., Spee E. J., Carmichael G. R., Potra F. A., Benchmarking stiff ode solvers for atmospheric chemistry problems II: Rosenbrock solvers**,** Atmospheric Environment, (31), 3469-3472, 1997

7.25: Please clarify how the increment vector is compared to the converge threshold (a scalar). Presumably some norm of the increment? State which norm and provide some justification for associated threshold value.

Convergence is deemed to have been achieved when the maximum fractional increment over all species calculated within the layer being considered by the solver is less than $1x10^{-4}$. This threshold has been the same for all versions of UKCA from at least UM vn6.1 onwards.

I can see the distinction that the authors are making between their method and typical QN, but it's a rather small difference.

It is certainly a small difference in theory, but in practice it makes a big difference when one is to choose between using purely QN steps after the initialisation or using QN selectively at the top of

NR steps. For non-stiff systems QN alone, despite being an approximation, can be used repeatedly and can be more efficient than using mixed method as here. But for our very *stiff* system, using only QN steps will almost certainly fail, or at least take longer than a pure NR approach. Therefore this distinction becomes vitally important and is a key aspect of our developments.

12.12: Seems like replacing NR iterations with QN isn't going to always reduce the overall computational cost, as the NR step is less accurate. Does it ever (or could it ever) occur that so many more NR iterations are required that the total cost is increased?

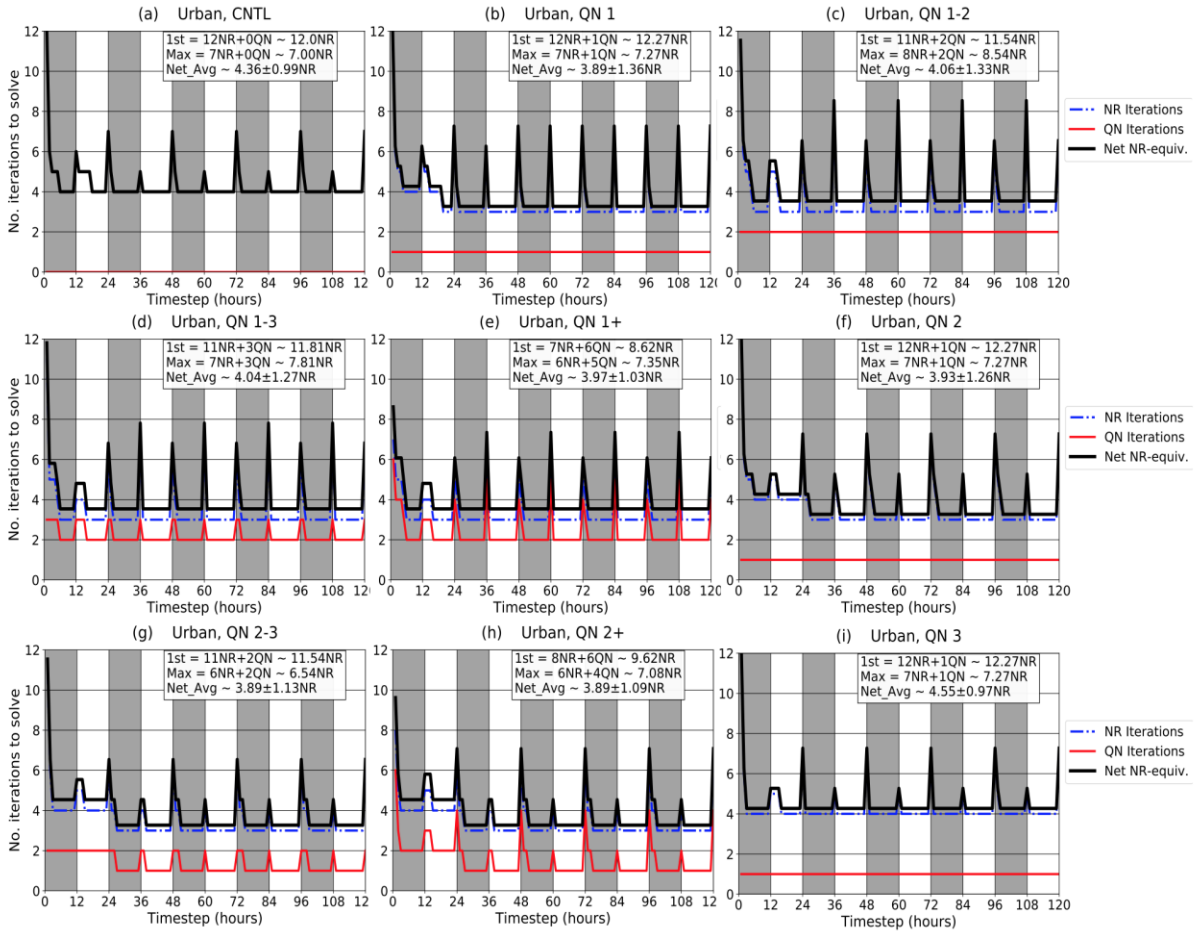
Yes, absolutely. This is what is meant by "diminishing returns" of increased number of QN iterations. For example, if you compare the QN2-3 scenario with QN2+ (Figure 4 g and h), at the dusk timesteps both scenarios take 6 NR iterations to solve (compared to 7 in the control run), however the QN2-3 uses a QN iterations twice (after the 2nd and 3rd NR iteration), whereas QN2+ called QN steps 4 times (after every NR iteration except the first and last). This means the QN2+ run has called the QN method two more times without reducing the number of NR iterations, such that the total cost is increased. The following sentence was extended to highlight this point:

"However, the QN method is not as exact as the NR method, and so there is not a one-to-one efficiency: calling the QN method many times may only reduce the number of NR iterations required by a few, and in some cases calling the QN method too many times can result in a net increase in computational burden."

This problem is why the different options for calling the QN iteration where tested, and the QN2-3 chosen as the best compromise in the widest range of situations.

Fig 4: Might also be useful to show the total computational expense, rather than # of iterations, as a function of timestep. Computational expense could be plotted on the right hand side axis.

Thank you, this is a good idea to more clearly highlight the take-home messages from this figure. The Figure has been edited to include a third solid black line of the net "Newton-Raphson equivalent" iterations (NR + 0.27*QN). The text discussing the figure has also been modified accordingly. This method has been used as an approximation of computational rather than actual computational expense because there is a huge amount of random variation between runs and with each call to the routine, such that the model needs to be run many thousands of times to get statistically robust average computational times (which is how the value of a QN iteration taking 27% of the tie of a NR step was derived originally).



"Figure 4. Plots of solver iteration (convergence) numbers for the original full Newton-Raphson (NR) method and Quasi-Newton (QN) methods, with QN pseudo-iterations only called on particular iteration(s). The CNTL scenario (top-left) only solves with NR iterations, and is equivalent to the solver in the release version of UKCA. The other 8 panels call QN pseudo-iterations on one or more iterations at each timestep. The blue dashed lines show number of NR iterations required to converge on a stable solution, the red line shows number of QN pseudo-iterations required, and the black line total net number of NR-equivalent iterations to solved, calculated as NR + 0.27QN. The white bands show periods with photolysis on, and grey band periods with photolysis off. The text in each panel gives the number of NR and QN iterations required to converge on the first timestep, the most difficult timestep after the first, and on average across the whole period in NR-equivalent iterations."

15: I was a little confused as to why QN2-3 was selected based on Fig 3 and Fig 4. It seems that in terms of overall computational expense (Fig 4), there are several where the net_Avg is about 3.9 NR. I would think then these methods should be evaluated in terms of overall accuracy (Fig 3) in a comprehensive manner, using a metrics such as significant digits of accuracy ,
SDA = -log10(max_k E_k)
where E_k is the root mean square norm of the relative error compared to the CNTRL simulation, and max_k indicates that the species with maximum E_k is considered. Then the algorithm with the fastest, most accurate, performance selected.

Setting up a box-model scenario that is representative of the conditions found in the global model is actually quite challenging, and the net average is not particularly representative. The decision of which scenario to use was taken more on the basis of how many iterations were required on the 'Max' timestep than the average (i.e. how it performs in the transition between light and dark). The reason for this is because in the global model, it can only proceed once all cores have completed their chemistry (see response to Reviewer #2 comment #1). As computing chemistry at dawn/dusk are typically the most challenging periods to solve, and as there will always be processors along the terminator somewhere in the world, the time taken to solve the "Max" step was considered more representative of the runtime of the global model than the simple average. Scenario QN2-3 has the lowest "Max" value in Figure 4 compared to all the other scenarios. The following discussion has been added to highlight this point:

"While the UKCA_BOX model only solves a single case at any one timestep, each core in the 3D model will solve for many gridcells at each timestep, and can only move on to the next timestep once all have converged. In other words, the 3D model is only as fast as its slowest gridcell. For this reason, the cases where the new methods reduce iteration count at the more challenging timesteps (at dawn and dusk) are considered a stronger indication that they will improve integration time in the full 3D model than the average."

Regarding the accuracy, the error of all of the scenarios were considered small enough to not be of consequence in making the decision. Errors in OH were some of the highest, and still typically $< 10^{-5}$ (SDA ~ 5) for all scenarios (Figure 3f). The largest SDA for all species was for OClO in the QN1+ scenario, with RMSE = 3.47^{-05} and SDA = 4.5. Given how this development has been coded, one would not expect the maximum error in any method to be more than 10^{-4} (SDA=4), because the same test is used to exit the solver in every case (when the maximum increment over all species is less than 10^{-4} after a Newton-Raphson iteration). The tolerance of 10^{-4} is the same as that used by default in the global model. We would be concerned if any of the experiments showed an error $>10^{-4}$ and/or diverged over time relative to the control, and none of the scenarios show this.

17.25: "did not bit" – typo?

This should read "did not bit-compare". To clarify this statement we have amended to text to:

"This means that the wind fields in these simulations were not identical as the small concentration changes introduced by the QN method resulted in global changes to the dynamical fields."

General: The authors might consider the applicability of their work to the field of chemical transport models, which spend a much greater fraction of their wall time on solving chemistry (since transport and dynamics are not solved online).

We thank the reviewer for this point. Indeed, we feel that the our hybrid approach would provide a bigger saving for models that are not calculating full atmospheric physics calculations and as

we have shown with the UKCA_BOX results, significant speed ups can be realised. This point has been added to the conclusions:

"If implemented in a chemical transport model, for example, one would expect the overall benefit to be greater, due to the greater proportion of computational expense of the chemical solver due to the lack of other online physical processes."

We thank the referee for his/her useful comments.

In addition to changes suggested by the referees we also made few small changes such as fixing typos and small textual changes to improve clarity which are all indicated/highlighted in the revised document (line numberings are according to the original document).

IN THE MANUSCRIPT

- *Pg 1 ln 2, Replaced "Vn" with "vn" in the heading*
- *Pg 1, the authors' affiliations have been clarified*
- *Pg1 ln 3, added the first name Nathan for Luke Abraham*
- *Pg 2 ln9, moved the Morgenstern reference slightly above*
- *Pg 4 ln 2, reworded the sentence as "...* generated within the iterations. The method is based on exploiting this information in a way that enable one*…"*
- *Pg 4 ln 16, added the text "with respect to number of main NR iterations"*
- *Removed commas on pg 5, ln 15 and 23. Pg 28 ln 15.*
- *Pg 6 removed the whole blank line before Section 2.1*
- *Deleted 'a' pg 6 ln 23*
- *Pg 7 before Section 2.3, added the text "the vector of species concentrations at the next chemical time step"*
- *Pg 7 ln ln 13, added the text "(or simply J^k)"*
- *Pg 7 ln18, replaced H with H^k and replaced the text (–J)^(-1) with (-J^k))^-1*
- *Pg 7 ln 22 Clarified type of initial predictor (forward Euler).*
- *pg 7 ln 25, Clarification the convergence is tested by a "relative change":*
- *Added commas: pg 10 ln15, ln 24. Pg 28 ln 27*
- *Pg 21, 22 and 23, Figures 5 and 6: clarified "iteration numbers" refers only to number of Newton-Raphson iterations (NR iterations).*
- *Moved "is" in sentence, pg 20, ln 2*
- *Pg 26 ln 19, (conclusion) we added the text "using options defined in the namelist"*

IN THE SUPPLEMENT

- *Pg 1, changed BoxModel->BOX_MODEL, FN iterations-> NR iterations*
- *Figures S2, S4, S6 are updated in the same way as in the main text*
- *Pg 11, caption of Figure S8, "OH" is replaced with "Ozone"*
- *Pg 12, added a new subsection containing the NMAD, NRSMD, NMB values of all species*

Quasi-Newton Methods for Atmospheric Chemistry Simulations: Implementation in UKCA UM ~~Vn10~~vn10.8

Emre Esenturk[1,2], Nathan Luke Abraham[1~~2~~,3], Scott Archer-Nicholls[1~~2~~], Christina Mitsakou[1~~2~~,b], Paul Griffiths[1~~2~~,3], Alex Archibald[1~~2~~,3], John Pyle[1~~2~~,3]

[1]~~Mathematics Institute, University of Warwick, Coventry, CV4 7AL, UK~~[1]Department of Chemistry, University of Cambridge, Cambridge, CB2 1EW, UK
[2]Mathematics Institute, University of Warwick, Coventry, CV4 7AL, UK
[2]~~Department of Chemistry, University of Cambridge, Cambridge, CB2 1EW, UK~~
[3]National Centre for Atmospheric Science, Cambridge, UK
[b] Currently at the Center for Radiation, Chemical Environments and Hazards, Public Health England, Chilton, Oxon, OX11 0RQ, UK

Correspondence to: *Emre Esentürk*

Abstract. A key and expensive part of coupled atmospheric chemistry-climate model simulations is the integration of gas phase chemistry, which involves dozens of species and hundreds of reactions. These species and reactions form a highly-coupled network of Differential Equations (DEs). There exists orders of magnitude variability in the lifetimes of the different species present in the atmosphere and so solving these DEs to obtain robust numerical solutions poses a "stiff problem". With newer models having more species and increased complexity it is now becoming increasingly important to have chemistry solving schemes that reduce time but maintain accuracy. While a sound way to handle stiff systems is by using implicit DE solvers, the computational costs for such solvers are high due to internal iterative algorithms (e.g., Newton-Raphson methods). Here we propose an approach for implicit DE solvers that improves their convergence speed and robustness with relatively small modification in the code. We achieve this by blending the existing Newton-Raphson (NR) method with Quasi-Newton (QN) methods, whereby the QN routine is called only on selected iterations of the solver. We test our approach with numerical experiments on the UK Chemistry and Aerosol (UKCA) model, part of the UK Met Office Unified Model suite, run in both an idealized box-model environment and under realistic 3D atmospheric conditions. The box model tests reveal that the proposed method reduces the time spent in the solver routines significantly, with each QN call costing 27% of a call to the full NR routine. A series of experiments over a range of chemical environments was conducted with the box-model to find the optimal iteration steps to call the QN routine which result in the greatest reduction in the total number of NR iterations whilst minimising the chance of causing instabilities and maintaining solver accuracy. The 3D simulations show that our moderate modification, by means of using a blended method ~~on~~ for the chemistry solver, speeds up the chemistry routines by around 13%, resulting in a net improvement in overall run-time of the full model by approximately 3 % with negligible loss in the accuracy. The blended QN method also improves the robustness of the solver, reducing the number of grid cells which fail to converge after 50 iterations by 40%. The relative differences in chemical concentrations between the control run and that using

the blended QN method are ~~negligible~~ of order $\sim 10^{-7}$ for longer lived species, such as ozone, and below the threshold for solver convergence (10^{-4}) almost everywhere for shorter lived species such as the hydroxyl radical.

1 Introduction

With the advent of supercomputers, simulating the atmosphere using computational models has become an integral part of atmospheric science research, complementing experimental measurements, *in-situ* and remote observations. Model predictions are playing an increasingly important role in both purely scientific investigations and public policy making (IPCC, 2013; Glotfelty et al., 2017). In recent years, increasing computational power has enabled the development of coupled chemistry-climate models (Morgenstern et al., 2009) which determine the chemical evolution and transport (Lauritzen et. al, 2009) of trace atmospheric constituents, such as long-lived greenhouse gases, ozone, nitrogen oxides, volatile organic compounds and aerosol particles, and their influence on the environment, air quality and human health (~~Morgenstern et al., 2009;~~ Heal et al., 2013; Lamarque et al. 2013; O'Conner et al., 2014; Tilmes et al., 2015; Collins et al., 2017). These models require globally accurate predictions over time frames that span decades (Lamarque et al. 2013), involving chemical reactions of species with lifetimes ranging from sub-seconds to centuries (Whitehouse et. al., 2004), making the task computationally very expensive.

The UK Chemistry and Aerosols (UKCA) model is part of the Met Office United Model (UM) (Hewitt et al, 2011) and works as its chemistry (Morgenstern et al., 2009, O'Connor et al., 2014) and aerosol (Mann *et al.*, 2010) component. Hereafter we refer to UM-UKCA as the fully coupled chemistry-climate model and make reference to the individual sub modules as UKCA and UM. Solving the chemistry in UKCA comes at a significant cost as it is one of the most expensive components in the UM-UKCA model. As coupled chemistry-climate models become more complex and the description of chemistry more involved, the need for computationally economic methods will be in higher demand. Hence, it makes sense to investigate ways of increasing the speed of the existing schemes with the goal of little or no sacrifice in accuracy.

Problems of a similar kind appear in other fields such as combustion systems which contain possibly reduced physical dynamics but more intensive chemistry (up to thousands of reactions) (Lu et. al., 2009) and aerosol microphysics and dynamics (Mitsakou et al, 2005). Mathematically these systems are represented by complex networks of coupled differential equations (DEs) which one must solve numerically. There is no universally best numerical method that works for every type of DE. Often one needs to choose the most reasonable method according to the need (e.g. ease of incorporating/modifying in model, solution CPU cost/time, accuracy). The numerical methods available can be conveniently categorized as explicit or implicit. Explicit methods are ~~quick and~~ direct integration methods that work for many types of conventional problems but have worse stability properties, while implicit methods are more involved and indirect in calculations but have superior stability properties (Atkinson, 1989; Sandu et. al 1997; Damian et. al, 2002). Generally, explicit methods are quicker than implicit methods at integration of single iteration step but can fall behind in the total integration cost due to the extra efforts to ensure stability (generally by halving the time steps). When it comes to atmospheric chemistry calculations, the main stumbling block against getting stable solutions is the problem of stiffness, which, broadly speaking, originates from different chemical reactions having orders of magnitude different time-scales (Cariolle et. al., 2016). If one uses an explicit DE method, the (approximate)

concentration values of the next timestep are calculated based on the tendencies at the current time. This makes it extremely hard to choose a timestep which is short enough to capture the chemical changes and preserve stability but also long enough to make the calculations feasible for computers. A good way to overcome this difficulty is by using an implicit method where tendencies are not based on current values, but treated as unknowns to be solved (along with the new concentration values). This greatly increases the stability of solutions at the cost of a series of extra calculations for each timestep. But again, there is no single best implicit method which is suitable to all types of stiff problems. In fact, there are families of numerical schemes available for each category (Atkinson 1989). It is therefore desirable for any proposed new method to be flexible enough so that they can be appended to the existing solver algorithms without substantial change. This is the aim of the proposed method here.

As will be detailed further in the text, a common feature of the many currently available implicit schemes is the solution of large systems of nonlinear differential equations iteratively (Ortega and Rheinboldt, 1970; Brandt, 1977; Kelley, 1995). At each timestep, expensive subroutines have to be called several times; this is the main source of computational cost of the chemical time integration. These subroutines typically include (i) construction of a Jacobian (derivative of a function in higher dimensions) (ii) a Newton-Raphson type iterative algorithm to solve the nonlinear algebraic equations (associated to the nonlinear differential equations). To overcome the high costs, methods that avoid or reduce Jacobian construction have gained popularity in recent years (Brown and Saad, 1990; Chan and Jackson, 1984; Knoll and Keyes, 2004; Viallet et al., 2016 and the references therein). Our motivation for this work is somewhat similar in that we use approximations of the Jacobian to reduce the costs of the solver.

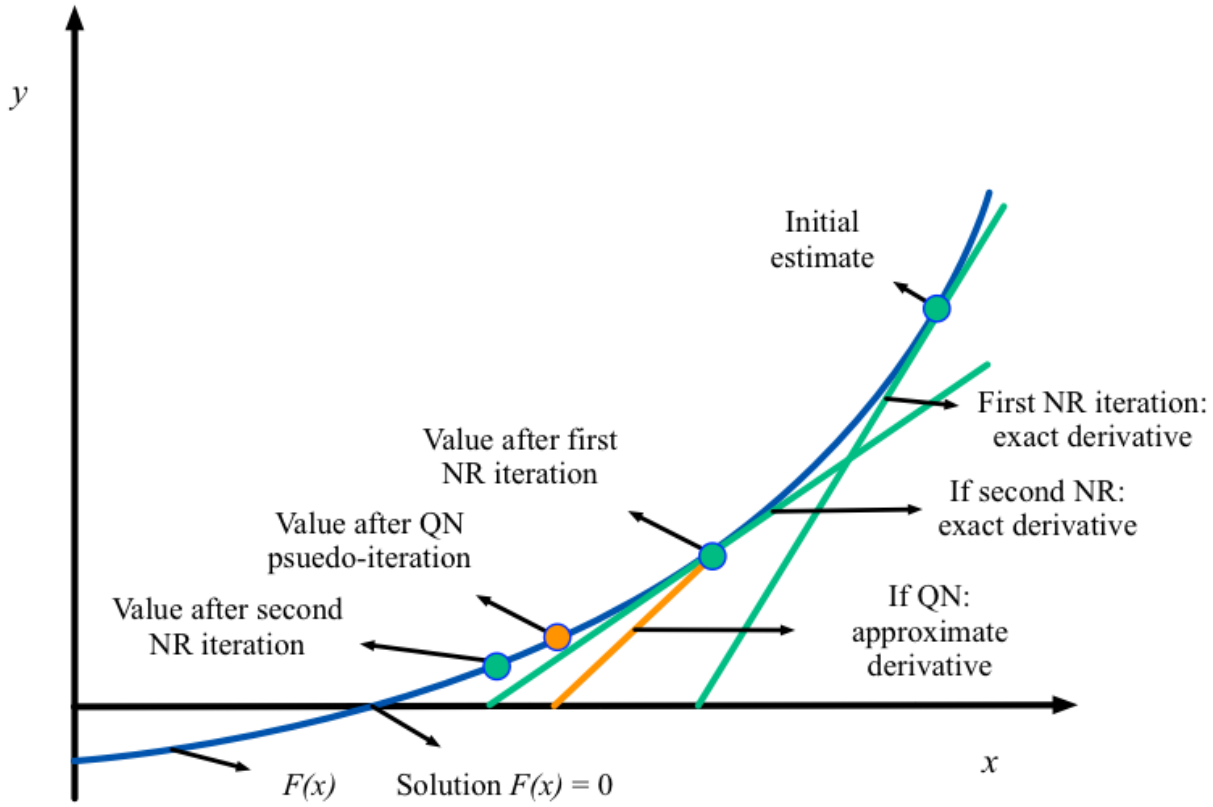


Figure 1: Illustration of application of the QN method (adopted in our work) to find the root of a function of one variable.

Here we develop an approach which reduces the costs of expensive routines by partly recycling the information ~~obtained~~ generated within ~~previous~~ the iterations. ~~We~~ The method is based on exploiting this~~em~~ information in a way that enable one~~us~~ to take extra steps forward for the desired solution without going through the costly parts of the cycle. The approach is an adaptation of the Quasi-Newton (QN) methods (Broyden, 1965; Shanno, 1970; Fletcher, 1970; Goldfarb, 1971; Davidon, 1991) fused into the classical Newton-Raphson (NR) method, which are commonly used for solving large systems of nonlinear algebraic equations.

The main idea behind the QN method is illustrated in Figure 1. The objective of finding species concentrations after a short time interval can be transformed into finding the roots of a nonlinear function, which, in Figure 1, is represented as a function $F(x)$ of a single variable x. Numerically, the task of finding the root of the function can be achieved by the NR algorithm which is based on finding the x-intercepts following the tangent lines of values of the function (the green lines in Figure 1). The root is obtained by simply re-evaluating the function at each x-intercept and iterating the process. The QN method uses an approximation for the tangent line (instead of an exact derivative), the orange line in Figure 1, so that computing the "new" x-

intercept is quicker. In higher dimensions (e.g when solving for multiple chemical species), finding the exact derivative is equivalent to calculating the Jacobian matrix, while the QN method uses an approximate Jacobian, saving considerable computation time. A key point of the implementation is that, the additional internal QN iterations do not replace the NR iterations completely. Rather each QN iteration works in and is fed by the current NR iteration.

Our adaptation of the QN method uses an 'inverse update' approximation (Kvaalen, 1991) instead of the more commonly used 'forward updates' (Broyden, 1965). We demonstrate that the approach improves the convergence rate significantly with respect to the number of main NR iterations and saves computational time. We further argue that using our mixed-method approach makes the algorithm more robust against "stiff environments" as it reduces the probability of the solver failing to converge on a solution and restarting using a shorter timestep. We also test how the solutions (chemical concentrations of species) are affected over a long period of integration. We show that the differences in prognostic variables between our suggested QN method and the classical NR method are negligible and do not grow in time.

The structure of this article is as follows. In Section 2 we describe the UM-UKCA model and give a brief summary of its basic features. We then outline the current algorithm that handles the reaction kinetics by solving systems of non-linear ordinary differential equations (ODE) followed by our suggested modification using Quasi-Newton methods. We further discuss why and how this modification works, its advantages and its possible dangers. In Section 3, we report results of our computational experiments carried out under both, a controlled box-model environment and as part of the full 3D Met Office UM-UKCA model. We compare the results of the code-modified runs with the control runs from the perspective of computational savings and differences in the concentrations/mixing ratios of chemical species, and discuss related matters with regard to parallel computing clusters. In Section 4 we conclude the paper by summarizing and highlighting our results and pointing to possible future directions.

2 The UKCA Model

UM-UKCA, originally developed by the National Centre for Atmospheric Science, and the UK Met Office, was designed as a framework for atmospheric chemistry and aerosol computations that operates under the Met Office United Model (UM) platform and models atmospheric chemistry and aerosol fields that can feed-back onto the model dynamics via the model radiation scheme (Morgenstern *et al.*, 2009; O'Connor *et al.*, 2014). It computes a number of possible physical-chemical processes taking place in the atmosphere such as radiation, photolysis, emissions, wet/dry deposition and clouds. It is coupled to the UM transport dynamics sequentially, that is, transport routines and chemistry-aerosol routines are performed one after another (operator splitting) with adjustable frequency. Currently in its global configuration, for transport a timestep of 20 min is used, whilst a chemical timestep of 1hr is used to update the new concentrations of species in the model.

A number of chemical schemes are available in UKCA for modelling different parts of the atmosphere (troposphere, stratosphere etc.) with varying model details (e.g. radiative feedback switched on/off). In this paper, we use the more general

stratospheric–tropospheric coupled scheme with and without an online aerosol mode (either using GLOMAP MODE (Mann et al., 2010) or aerosol climatologies) to demonstrate our results. The pure stratospheric–tropospheric mode (StratTrop) contains 75 species and consists of 283 chemical reactions (Banerjee et al., 2016). When GLOMAP MODE aerosols are activated, 12 additional tracers are added to the system and a total of 306 reactions represent the atmospheric chemistry. The StratTrop chemical mechanism is solved using an implicit backward Euler scheme under the ASAD framework (Carver et al., 1997; Wild and Prather, 2000), as described in detail below, while photolysis is computed using the FastJ-X scheme (Wild et al., 2000). The details of these schemes can be found in Abraham et al. (2012). The UM-UKCA version used here is vn10.6.1, in the Global Atmosphere 7.1 configuration, which is a development of the UM-UKCA GA6 configuration (Walters et al, 2016).

In addition to the full 3D UM-UKCA model, we also use a box-model version of UKCA (hereafter referred as UKCA_BOX) to gain better control of the chemistry part of our simulations. UKCA_BOX is designed as a development tool using the same UKCA code, branched from version 10.1 of the UM-UKCA, but with the rest of the UM-UKCA model removed and replaced with inputs that feed the UKCA code with the same information as if it were a single grid cell in the full 3D model. The box model uses the same StratTrop (CheST) chemical mechanism, ASAD chemical solver and FastJ-X photolysis scheme as the full 3D model, but does not have any emissions, deposition or transport. As it runs for only a single grid cell, it can be run cheaply on a single processor across many test cases. Thus, it is ideal for testing and optimising the chemical solver in UKCA over a wide range of idealised chemical environments.

In the following sections, we discuss the chemical time integration schemes in the UKCA package for determining the new tracer concentrations and chemical tendencies. All numerical schemes are implemented using the Fortran 95 language. The code is available in the UM-UKCA trunk from version 10.8. Branches are also available at vn10.7 and vn10.6.1.

2.1 Chemical evolution in the UKCA

The time integration for the gas-phase chemistry in UKCA is carried out by the ASAD package which provides a flexible framework for adding and removing new reactions/species (Carver et al., 1997; Wild and Prather, 2000). The UKCA version of the ASAD package uses a backward Euler numerical scheme to compute the new species concentrations at the next chemical time step. One of the reasons for this choice is that the relevant time scales of the reactions of species vary over many orders of magnitudes depending on the location and time of the reactions which makes the system extremely stiff. The backward Euler method is an implicit scheme which has superior numerical stability properties than almost all other explicit or semi-explicit methods and hence works particularly well with stiff systems (Atkinson 1989). This enables the use of longer timesteps and makes long time-integrations feasible. The drawback is that, as in all implicit schemes, it demands that systems of nonlinear

algebraic equations are solved at each time step, requiring extra calculations and so increasing the computational cost significantly.

These heavy costs can be partly reduced by exploiting the fact that the coupling among species is "loose" in the sense that each species reacts with several other species but not all. This makes the Jacobian sparse and allows for the use of sparse matrix methods which significantly cuts costs. This approach was implemented in the UM-UKCA model (see Morgenstern et al, 2010).

2.2 Numerical implementation in the existing solver

The reaction kinetics in the atmosphere can be represented, mathematically, as a system of nonlinear ODEs where the initial values are prescribed. Emissions and dry/wet deposition enter in these equations as source and sink terms. The task of determining the change in ~~a~~ chemical species concentrations is equivalent to solving the coupled nonlinear system numerically.

Let $\boldsymbol{c}(t) = (c_1(t), c_2(t), \dots, c_N(t))$ denote the vector of species concentrations at a given time. Then the species evolve according to

$$\frac{d\boldsymbol{c}}{dt} = \boldsymbol{f}(\boldsymbol{c}) = \boldsymbol{P}(\boldsymbol{c}) - \boldsymbol{L}(\boldsymbol{c}) + \boldsymbol{E} - \boldsymbol{D_{wet}}(\boldsymbol{c}) - \boldsymbol{D_{dry}}(\boldsymbol{c}) \quad (1)$$

$$\boldsymbol{c}(0) = \boldsymbol{a}, \quad (2)$$

where **f** is the non-linear vector function (tendencies) given by the production and loss terms $\boldsymbol{P}$, $\boldsymbol{L}$, emissions $\boldsymbol{E}$, and wet and dry depositions $\boldsymbol{D_{wet}}$, $\boldsymbol{D_{dry}}$. The vector **a** is the initial concentration. The variables in bold-italic font are understood to be vectors. ~~–~~In the current implementation, emissions are treated separately during the boundary-layer mixing step, and dry deposition occurs throughout the boundary layer.

To solve Equation (1) numerically using a backward Euler scheme we discretize the time variable, so the discrete equation takes the form

$$\frac{\boldsymbol{c}(t_* + \Delta t) - \boldsymbol{c}(t_*)}{\Delta t} = \boldsymbol{f}(\boldsymbol{c}(t_* + \Delta t)), \quad (3)$$

where t_* is the current time and Δt is the difference between the next chemical timestep and current time. The unknown $\boldsymbol{c}(t_* + \Delta t)$~~-~~, the vector of species concentrations at the next chemical timestep appears on both sides of the nonlinear equation which can be solved numerically using a Newton-Raphson (NR) algorithm.

2.3 Newton-Raphson (NR) scheme

Here we give a brief description of the NR method, which will prepare the ground for discussion of our contribution. Setting $t = t_* + \Delta t$ and $\boldsymbol{c}(t_*) = \boldsymbol{c}_*$ for brevity, we first write the discretized ODE (Eq. 3) in the standard form of an algebraic equation (AE), that is,

$$\boldsymbol{F}(\boldsymbol{c}(t)) = \frac{\boldsymbol{c}(t)-\boldsymbol{c}_*}{\Delta t} - \boldsymbol{f}(\boldsymbol{c}(t)) = 0. \quad (4)$$

The NR scheme starts with an initial guess (e.g. solution from the previous time step or a first order predictor) followed by an iteration algorithm in which the following system of linear equations is solved,

$$\mathbf{J}(\boldsymbol{c}^k)(\Delta\boldsymbol{c}^k) = -\mathbf{F}(\boldsymbol{c}^k), \quad (5)$$

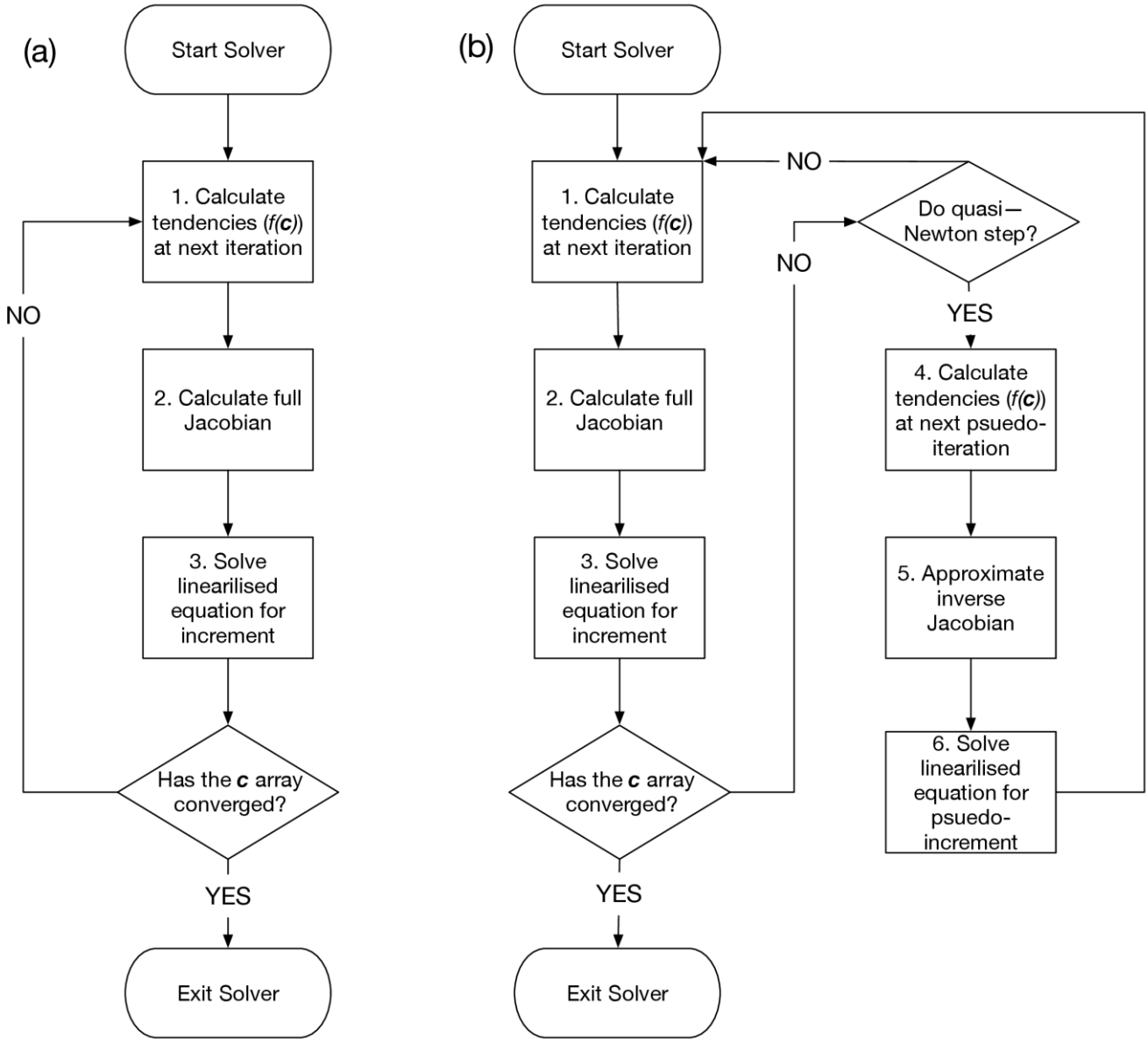
where $\mathbf{J}(\boldsymbol{c}^k)$ (or simply $\mathbf{J}^k$) is the Jacobian at the kth iterate and $\Delta\boldsymbol{c}^k = \boldsymbol{c}^{k+1} - \boldsymbol{c}^k$ is the increment (still within the same chemical timestep). At each iteration, by solving a linear equation of the form of Eq. 5, our initial guess will be improved and approaches to the actual solution of Eq. 4 as the procedure is repeated (Atkinson, 1989).

The linear equation (Eq. 5) can also be written in the form

$$(\Delta\boldsymbol{c}^k) = \mathbf{H}(\boldsymbol{c}^k)\boldsymbol{F}(\boldsymbol{c}^k), \quad (6)$$

where $\mathbf{H}(\boldsymbol{c}^k)$ (or simply $\mathbf{H}^k$~~$\mathbf{H}$~~) is the negative of the inverse of the Jacobian $(-\mathbf{J}^k)^{-1}$~~$(-\mathbf{J}^{-1})$~~. This form will be particularly useful when we explain our improvement of the current method.

In the current UKCA implementation each major calculation step of the ODE solution algorithm is carried out by a separate routine as shown in Figure 2a. The main solving engine begins by calculating the current tendencies (right hand side of Eq. 1) using the updated chemical concentrations from the previous timestep (Step 1 in Figure 2a). Then ~~A~~an initial predictor guess (forward Euler type) is~~ then~~ calculated to be used in the following iterative loop. ~~Then~~After that, the Jacobian is calculated using the exact quadratic form of the nonlinear reaction rates (Step 2). This step is followed by the solution of the linear Eq. 6 (Step 3). After the new increment ($\Delta\boldsymbol{c}^k$) is calculated, convergence is tested to determine whether $\Delta\boldsymbol{c}^k$is within our tolerance limit (which is set to a relative change of 10^{-4} in the current version). If the routine passes the convergence test, the solver exits and concentrations at the next timestep are output, otherwise the process repeats until it converges on a stable solution. If the solution fails to converge after a set number of iterations (50 in the current version), is unstable, or diverges, the routine will exit and repeat using a smaller time step (typically by halving the timestep). The expensive parts of the above procedure are, particularly, Step 2 and 3 (Figure 2a) and our goal is to reduce the number of calls to these steps as we show in the next section.


(a)
Start Solver
1. Calculate tendencies (f(c)) at next iteration
2. Calculate full Jacobian
3. Solve linearilised equation for increment
Has the c array converged?
NO
YES
Exit Solver
(b)
Start Solver
1. Calculate tendencies (f(c)) at next iteration
Do quasi—Newton step?
NO
NO
YES
2. Calculate full Jacobian
3. Solve linearilised equation for increment
Has the c array converged?
4. Calculate tendencies (f(c)) at next psuedo-iteration
5. Approximate inverse Jacobian
6. Solve linearilised equation for psuedo-increment
YES
Exit Solver

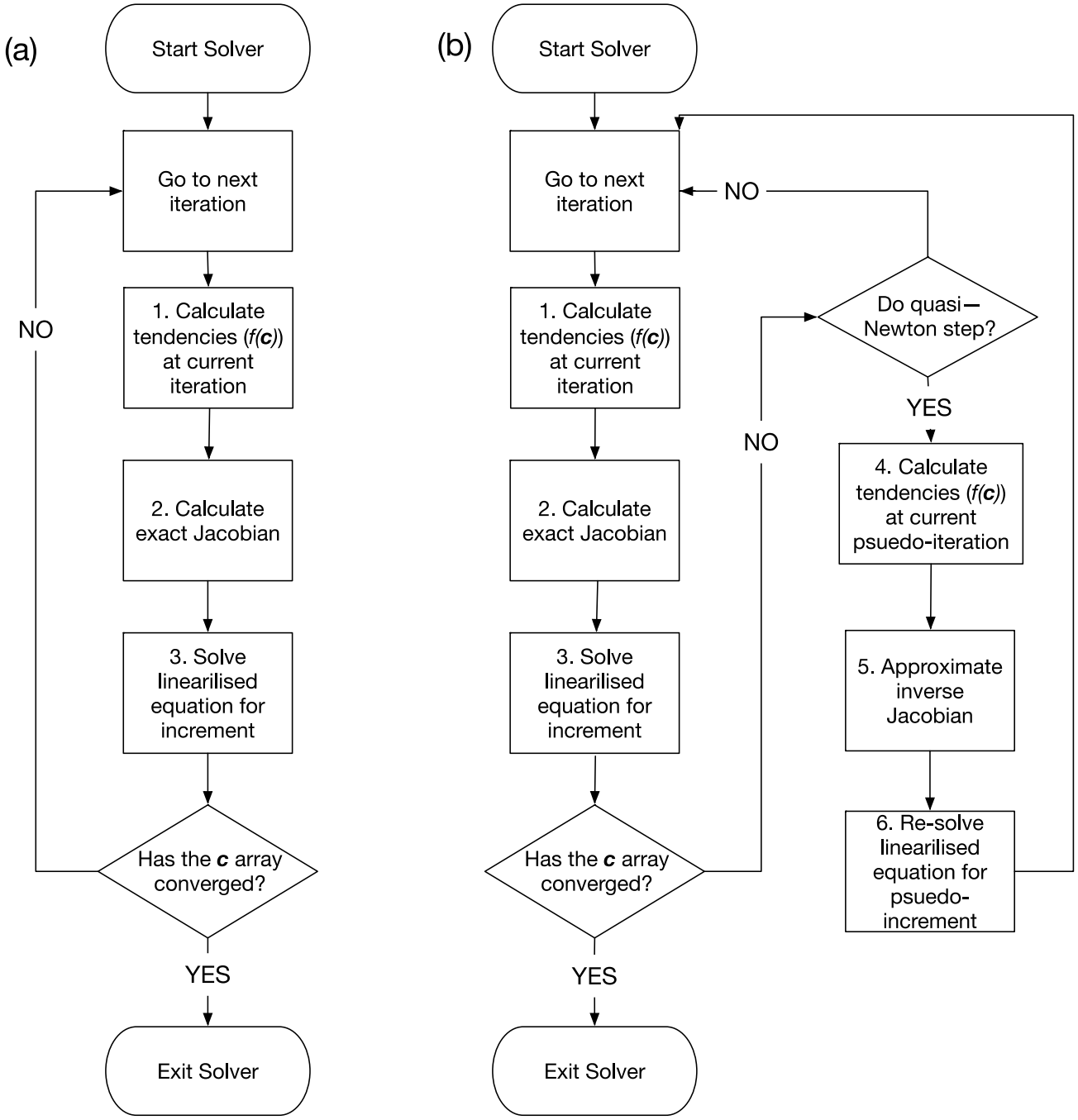


Figure 2: The flowchart: flowchart showing steps taken to numerically solve the non-linear chemical equations using the Newton-Raphson method; as carried out in the standard version of ASAD in the UKCA chemical transport model (a), and in our modified version incorporating a "Quasi-Newton iteration" (b).

2.4 Quasi-Newton (QN) Algorithm

We noted above that the expensive parts of the chemical integration are the Jacobian construction and solution of a system of linear equations at each iteration. Our strategy is based on the idea of using Quasi-Newton (QN) methods to minimise the number of iterations in the main Newton-Raphson (NR) solving loop, thereby reducing the number of Jacobian reconstructions and linear systems to be solved.

In QN methods, the use of exact Jacobian at every iteration is abandoned. Instead it is approximated it in a way that will satisfy certain imposed conditions. The ideas behind these (secant) methods, which date back to Broyden (1965), Shanno (1970), Fletcher (1970), Goldfarb (1971) and Davidon (1991) resemble using the inverse quotient of a function (of one variable) to replace the reciprocal of the exact derivative of the same function (see Figure 1). The price of this avoidance is a slowdown in convergence (not quadratic as in the NR algorithm, but still super-linear). In general, this strategy is more profitable since the slowdown in the convergence rate can be compensated by the substantial time gain obtained from bypassing the other costly steps compared to the time lost in the number of iterations.

Our implementation is somewhat different from the standard Quasi-Newton methods in that Newton-Raphson iterations are not completely replaced by the QN iterations. Rather, QN iterations are fused into the existing NR loop and implemented only if a chosen criterion is met. In this sense the new algorithm is a mixed method which uses both NR and QN methods as needed. This way keeps the changes to the existing algorithm minimal and makes the method flexible and practical to use. Despite this relatively small change in the algorithm the computational gain in return is considerable.

Diagrammatically (see Figure 2b), the approach works as follows: If the desired convergence has not taken place after the end of the Newton-Raphson iteration, then instead of moving on to the next iteration and reconstructing the Jacobian from scratch (Step 2), we make a pseudo-iteration and form an "effective approximation" for the inverse of the Jacobian using the concentrations already computed (Step 5). Step 6 follows in which we re-solve for the newer concentration values making use of the information available from Step 3. So, a full NR iteration is *effectively* replaced by a QN pseudo-iteration taking much less time. These measures are quantified in Section 3.1.

In the above description we refer to the "effective approximation" of the inverse of the Jacobian. However, in practice, we do not strictly construct an approximate "inverse" since taking the inverse of a matrix brings more expense. Rather, the remnants of the main NR iteration (the Jacobian from Step 2, concentrations from Step 3) are recycled and used in the approximation scheme for the inverse of the Jacobian (Broyden approximation). Schematically, after the main Newton-Raphson route we perform the following steps of 4-5-6 shown in Figure 2b which is formalized below.

We use a particular, Broyden type inverse approximation scheme (Kvaalen, 1991), which is given by the following form

$$\mathbf{H}_{\boldsymbol{app}}^{k} = \mathbf{H}^{\boldsymbol{k}} - \frac{\Delta \boldsymbol{c}^{k} + \mathbf{H}^{k}(\Delta \boldsymbol{F}(\boldsymbol{c}^{k}))}{(\Delta \boldsymbol{c}^{k})^{T}(\Delta \boldsymbol{c}^{k})} (\Delta \boldsymbol{F}(\boldsymbol{c}^{k}))^{T} \quad (7),$$

where $\Delta \boldsymbol{c}^k = \mathbf{H}^k \boldsymbol{F}(\boldsymbol{c}^k)$ and $\Delta \boldsymbol{F}(\boldsymbol{c}^k) = \boldsymbol{F}(\boldsymbol{c}^k + \Delta \boldsymbol{c}^k) - \boldsymbol{F}(\boldsymbol{c}^k)$ are the increments from the $\boldsymbol{k}^{th}$ main iteration step, $\mathbf{H}^k$ is the inverse of the (exact) Jacobian in the main step (at the $\boldsymbol{k}^{th}$ iteration) and $\mathbf{H}^k_{\boldsymbol{app}}$ is the negative of the approximate inverse of the Jacobian in the pseudo-iteration after the $\boldsymbol{k}^{th}$ iteration. The superscript " $\ldots^T$ "denotes the transpose of a matrix. Although the above relation requires us to know the inverse of the Jacobian, for our purposes, we do not need to compute it explicitly.

Once $\mathbf{H}^k_{\boldsymbol{app}}$ is determined, the new pseudo-increment $\delta \boldsymbol{c}^k$ is given by the relation

$$\delta \boldsymbol{c}^k = \mathbf{H}^k_{\boldsymbol{app}} \boldsymbol{F}(\boldsymbol{c}^k + \Delta \boldsymbol{c}^k) \quad (8).$$

Taking $\mathbf{H}^k_{\boldsymbol{app}}$ from Equation 7 and placing it into Equation 8 gives

$$\delta \boldsymbol{c}^k = \mathbf{H}^k \boldsymbol{F}(\boldsymbol{c}^k + \Delta \boldsymbol{c}^k) - (\Delta \boldsymbol{c}^k + \mathbf{H}^k(\Delta \boldsymbol{F}(\boldsymbol{c}^k))a^k,$$

where $a^k = \frac{(\Delta \boldsymbol{F}(\boldsymbol{c}^k))^T \boldsymbol{F}(\boldsymbol{c}^k + \Delta \boldsymbol{c}^k)}{(\Delta \boldsymbol{c}^k)^T (\Delta \boldsymbol{c}^k)}$. Now, recalling $\Delta \boldsymbol{F}(\boldsymbol{c}^k) = \boldsymbol{F}(\boldsymbol{c}^k + \Delta \boldsymbol{c}^k) - \boldsymbol{F}(\boldsymbol{c}^k)$ and using the linearity of $\mathbf{H}^k$ and noting that $\Delta \boldsymbol{c}^k = \mathbf{H}^k \boldsymbol{F}(\boldsymbol{c}^k)$ the terms simplify

$$\delta \boldsymbol{c}^k = \mathbf{H}^k \boldsymbol{F}(\boldsymbol{c}^k + \Delta \boldsymbol{c}^k) - \mathbf{H}^k \boldsymbol{F}(\boldsymbol{c}^k + \Delta \boldsymbol{c}^k) a^k = (1 - a^k) \mathbf{H}^k \boldsymbol{F}(\boldsymbol{c}^k + \Delta \boldsymbol{c}^k). \quad (9)$$

If compared, we see that Eq. 9 has the same form as Eq. 6, which can also be written in the form of Eq. 5 as

$$\mathbf{J}(\boldsymbol{c}^k)(\delta \boldsymbol{c}^k) = (1 - a^k) \boldsymbol{F}(\boldsymbol{c}^k + \Delta \boldsymbol{c}^k) \quad (10).$$

Now, crucially, the information of the reduced (row echelon) matrix obtained from the original Jacobian through Gaussian elimination in Step 3 (Eq. 5) is still available and can readily be used to solve the linear Eq. 10, where the only difference from Eq. 5 is only in the right-hand side. This bypasses the need for computing $\mathbf{H}^k_{\boldsymbol{app}}$ explicitly, saving memory and time. In effect, the method accomplishes two tasks at once, reducing the combined steps of reconstructing a new Jacobian and solving a new linear equation (in a new NR~~main~~ iteration) into a single step of solving a modified linear equation (in a pseudo-iteration) based on the information already available within that (main) iteration. In practical terms, this means that $\sim N^3$ numerical operations that are normally needed to solve a linear system is now reduced to $\sim N^2$ operations which gives substantial savings within the routine. An example of the implementation of these changes is given in the pseudo-code provided in Appendix A.

3. Numerical Results

In this section, we compare our results with the new method (Quasi-Newton) and without (Classical Newton-Raphson) when implemented on the current version of the UKCA solver. We consider the effectiveness of the algorithm on a single processor

with, i.e., UKCA_BOX, as well as on a High Performance parallel Computing (HPC) platform (ARCHER) with the full 3D UM simulations. In both cases our analysis will be two-fold: comparison of computational performance (savings, robustness, etc.) and comparison of predicted model values. We show that, although the chemistry step alone takes 5% to 10% of the entire computations, there is a noticeable speed up when the chemistry component is modified in the way suggested without causing any significant error in prognostic variable values. This also improves the robustness of the computation by reducing the number of cases during the course of entire chemical integration for which the timestep has to be halved in order to converge on a solution.

3.1 UKCA_BOX Simulations

To test the performance of the Quasi-Newton (QN) approximation method on performance of the UKCA chemistry solver, we first tested the changes in UKCA_BOX. UKCA_BOX allows us to test the performance of the QN methods under a highly-controlled environment, and optimise the options for the solver based on a variety of chemical conditions.

Four standard testcases were setup for these experiments to test the behaviour of the box model in different chemical environments: Urban, Rural, Marine and Stratosphere (Strat). The initial conditions for these testcases were extracted for July from a 10-year run of the full UM-UKCA model for the year 2000 at 1.875°×1.25° resolution, equivalent to the experiments conducted in section 3.2. For the Urban, Rural and Marine scenarios, average surface chemical fields, temperature, pressure and specific humidity were extracted at surface locations over the Beijing megacity, continental USA and the Pacific Ocean respectively (see Table- 1 for details)~~.~~. All UKCA_BOX experiments were run on a single processor core. The Strat scenario used zonally averaged chemical and meteorological fields at 40°N and 32km. Full details of the scenarios are given in the supplement. The Urban scenario is initialised with the most complex mix of chemical components, and is therefore the most challenging to solve. For this reason, the analysis in the paper will focus on the Urban scenario. Results from the other scenarios are included in the supplement.

Table 1. Summary of data points from UM model runs used to initialise UKCA_BOX scenarios, parameters describing atmospheric conditions of each scenario, and initial concentrations of select chemical species. In each case, data is extracted from a 10 year July average run of the UM-UKCA model for the year 2000.

Scenario	Location	Height above ground (km)	Pressure (hPa)	Temperature (K)	Specific Humidity (kg/kg)	O3 (ppbv)	NOx (ppbv)	HCHO (ppbv)
Urban	40ºN, 116.4ºE	0	983	300	0.0147	46.5	20.5	3.37
Rural	40ºN, 260ºE	0	926	304	0.0101	49.8	2.5	1.95
Marine	40ºN, 180ºE	0	1017	292	0.0121	25.0	0.30	0.39

Stratosphere (Strat)	Zonal average at 40ºN	32	8.61	240	3.44E-6	9,102	15.7	0.09

The UKCA_BOX uses the Fast-JX photolysis scheme (Wild et al., 2000), comparable to that used in the full UM-UKCA model (Telford et al., 2013). For the purposes of these experiments, a simplified setup was used whereby photolysis turns 'on' and 'off' every 12 hours of integration, using pre-calculated photolysis rates. This was done to minimise the computation of photolysis rates, and create idealised scenarios with an abrupt step-change at 'dawn' and 'dusk' to test the stability of the solver. Photolysis rates were taken from an offline run of the 1D column Fast-JX scheme at 12 noon on 1st July, 40ºN at 0 km and 32 km in clear-sky conditions for the Urban, Rural and Marine and Strat scenarios respectively. Each experiment ran for 5 days with a 60 minute timestep (the same as the chemical timestep used in the full 3D UM-UKCA model). Without emissions, deposition or transport, the chemical evolution is completely determined by the initial conditions. Each scenario starts in a state of disequilibrium, then slowly 'winds down' over the 5 days of integration.

As discussed in the previous section, the QN method is cheaper than the full Newton-Raphson (NR) method because it does not recalculate the full Jacobian at each iteration (Table 2). On average, one QN iteration takes 27 % of the time of a full NR iteration. Since the QN method reduces the number of NR iterations required to converge, the time taken will therefore generally be reduced. However, the QN method is not as exact as the NR method, and so there is not a one-to-one efficiency: calling the QN method many times may only reduce the number of NR iterations required by a few, and in some cases calling the QN method too many times can result in a net increase in computational burden. Finding the most efficient setup therefore becomes an optimisation problem: how can we gain the maximum reduction in NR iterations, with as few calls to the QN method as possible? In particular, we are interested in reducing the number of iterations required for the solver during the most challenging chemical states when the equations are most stiff. This will reduce the range of time taken for cores to solve each part of the domain, therefore reducing time spent waiting for all cores to catch up to the same time in the full 3D model.

Table 2. Wallclock times for running 1000 calls for the ~~FN~~ NR iterations and QN iteration within the UKCA_BOX ~~box~~ model run on a single processor core.

	Full Newton Raphson Method	Quasi-Newton Method	Ratio
CPU time for 1000 calls	160 ± 3.1 ms	42 ± 0.71 ms	0.2625
Wallclock time for 1000 calls	157 ± 1.8 ms	42.9 ± 0.15 ms	0.273

To test the range of options, we devised 9 experiments for each scenario, as summarised in Table 3. The control (CNTL) experiment does not call the QN method, and is identical to the solver in the release version of UKCA. The other scenarios call the QN method after one or more NR-iterations, as given by the numbers in the name of experiments in Table 3. For

example, QN1 calls the QN Newton method after the first NR iteration only, QN2-3 calls it after the second and third NR-iterations, and QN1+ calls the QN method after every NR iteration. In general, the first iteration of the solver is where the solution is most likely to diverge and cause stability problems, and so a dampening factor of 0.5 is applied to the QN method, as is also done on the first iteration for the NR method. As shown by the flow structure of this development (Figure 2b), the QN method is only called if the solution has not already converged.

Table 3. Summary of experiments conducted using UKCA_BOX. The control (CNTL) experiment does not call the QN method. The other experiments call the QN method after one or more NR iterations.

Call QN method on	CNTL	QN1	QN1-2	QN1-3	QN1+	QN2	QN2-3	QN2+	QN3
1st iteration:	No	Yes	Yes	Yes	Yes	No	No	No	No
2nd iteration:	No	No	Yes	Yes	Yes	Yes	Yes	Yes	No
3rd iteration:	No	No	No	Yes	Yes	No	Yes	Yes	Yes
>3rd iteration:	No	No	No	No	Yes	No	No	Yes	No

Figure 3 shows chemical concentrations for a selection of chemical tracers from the box model, comparing the CNTL experiment with the QN experiments, for the Urban scenario. Similar figures for the other scenarios are included in the supplement. In this scenario, the mix of NOx and VOCs results in production of O_3 for the first day, then a slow loss of O_3 over the next four days as concentrations of short-lived tracers decay due to the lack of fresh emissions (Figure 3a). Overall, these results show the QN method is very accurate with negligible divergence from the CNTL experiment. The fractional differences are largest for short-lived tracers, such as OH, but are at most of the order 10^{-5} or less (Figure 3f). For longer lived species, such as O_3 or NO_2, fractional changes are typically $<10^{-8}$ (Figure 3c, i). Differences between the CNTL and QN scenarios do not grow over time, rather they tend to be largest in periods which are challenging to solve (at the start of the simulation, and around dawn and dusk) and then to decay to zero.

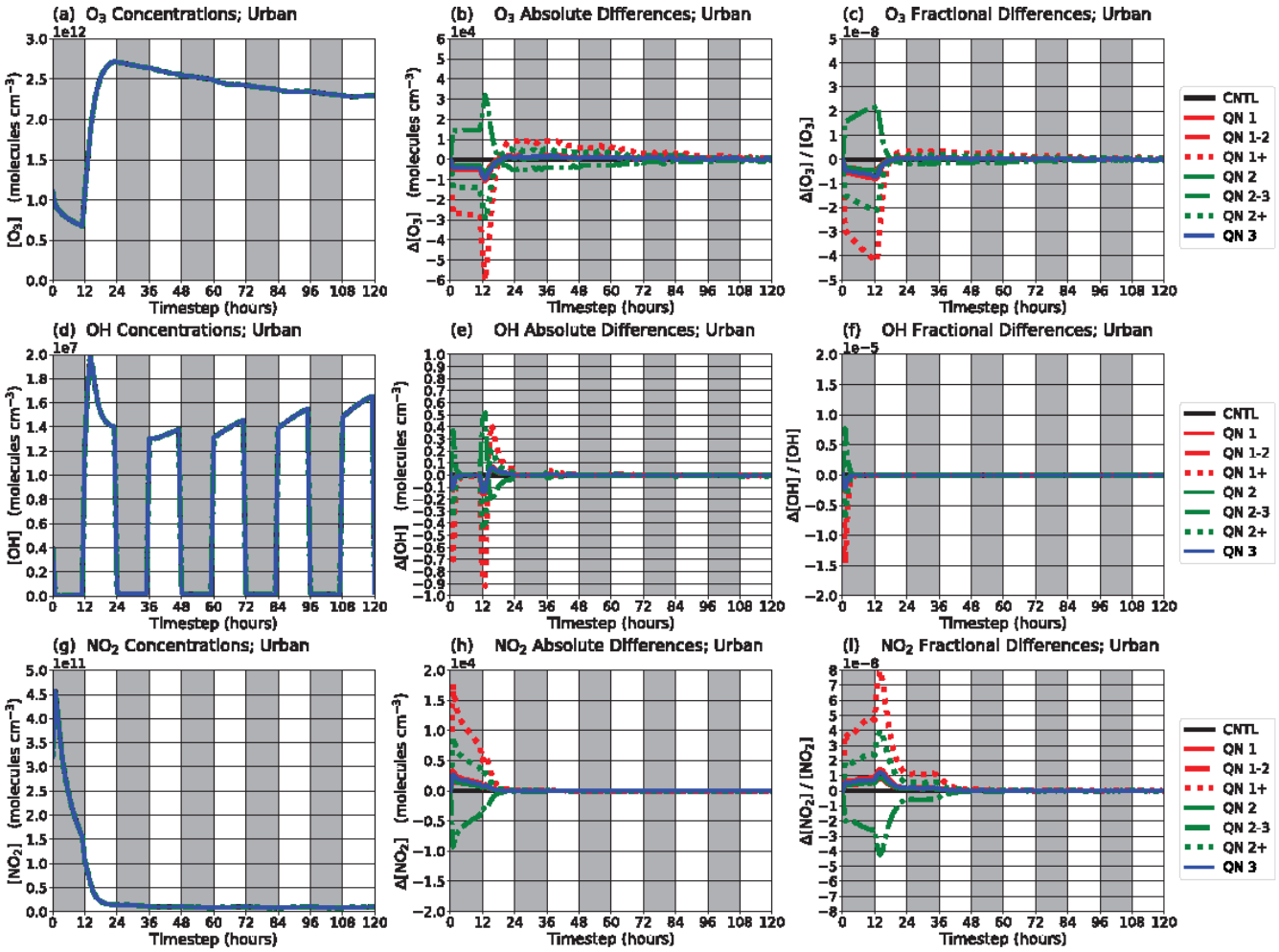
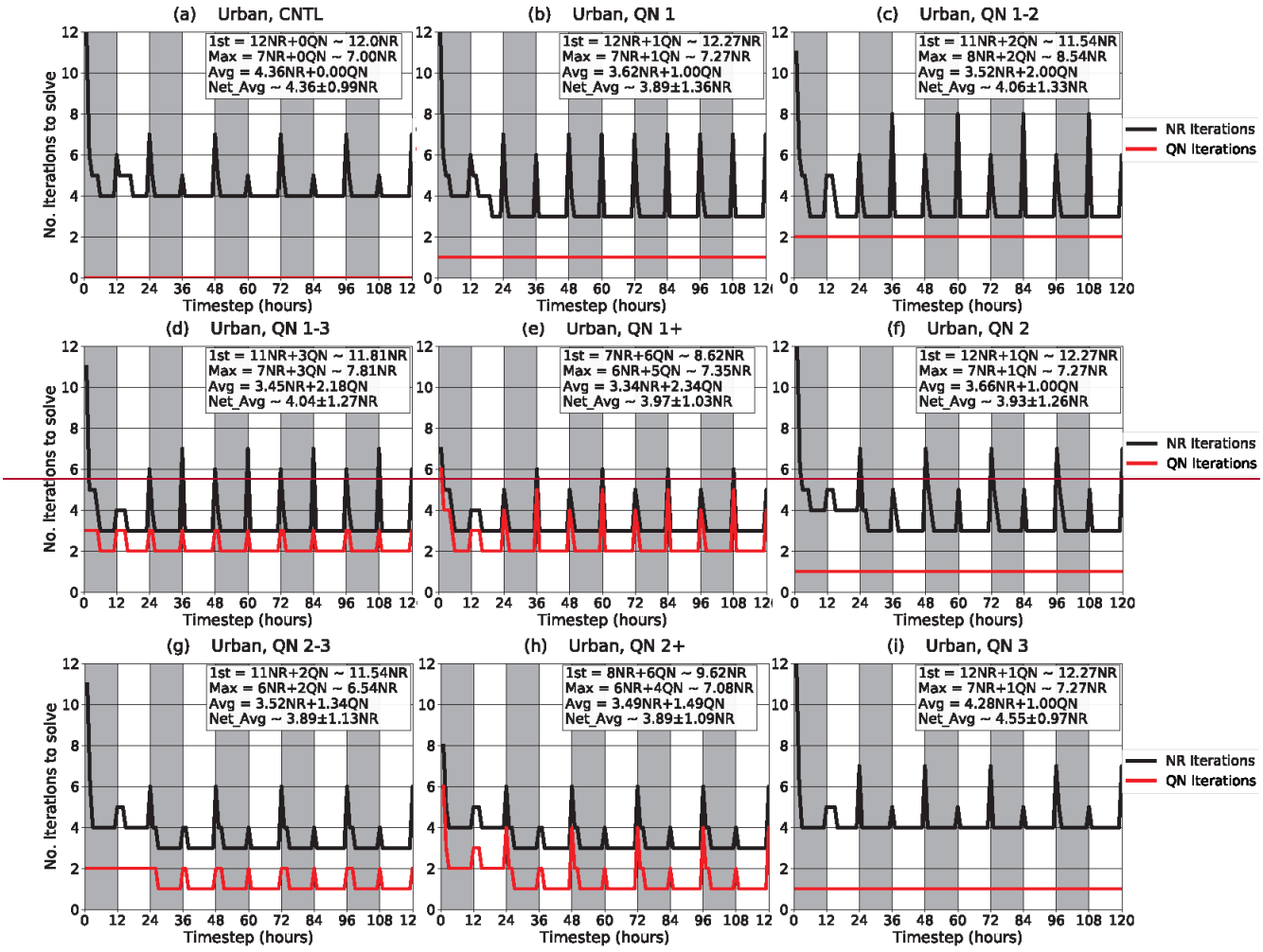


Figure 3. Concentrations of O_3, OH and NO_2 in molecules cm^{-3} from UKCA_BOX simulations of the Urban scenario. The left panels show absolute concentrations from all scenarios, with differences too small to be observed by eye. The centre and rightmost panels show absolute and fractional differences between the CNTL and QN experiments respectively. The white band show periods with photolysis on, and grey band periods with photolysis off.

Time series of the number of iterations required to converge for the Urban scenario are shown in Figure 4. Similar figures for the Rural, Marine and Strat scenarios are included in the supplement (Figures S1 and S2; Figures S3 and S4; Figures S56 and S6 respectively), which in general are found to converge in fewer iterations than the Urban case. ~~The black line shows the number.~~ The dashed blue line shows the number of NR iterations required to reach a stable solution at each timestep, and the red line shows the number of QN iterations required, and the black line the estimated NR-equivalent number of iterations taken to solve, using the result that QN iterations take on average 27% of the computational time to solve compared to the NR

method (Table 2). ~~.~~ The first timestep is ~~in general~~ the most ~~stiff~~difficult to solve, as ~~. T~~the initial chemical concentrations are typically far from a steady state having been taken from monthly average values from model cells. After that, t~~T~~he dawn and dusk periods, the timesteps immediately after photolysis is turned on and off respectively, are the next most challenging, as changing photolysis rates causes an abrupt change in the lifetimes of many species. The inclusion of the QN method can be seen to improve the solver when the ~~number of~~net NR equivalent iterations (black line) is lowered compared to the CNTL scenario, and is optimal when this can be achieved with the minimum number of QN pseudo-iterations (red line, Figure 4). While the UKCA_BOX model only solves a single case at any one timestep, each core in the 3D model will solve for many gridcells at each timestep, and can only move on to the next timestep once all have converged. In other words, the 3D model is only as fast as its slowest gridcell. For this reason, the cases where the new methods reduce iteration count at the more challenging timesteps (at dawn and dusk) are considered a stronger indication that they will improve integration time in the full 3D model than the average.

The Urban scenario is the most challenging of the testcases to solve, due to the high initial concentrations of reactive tracers (Figure 4). The CNTL scenario takes 12 full NR iterations to solve the first timestep, then between 4 and 7 for each timestep thereafter, needing 4.36 iterations on average (Figure 4a). More iterations are required at dawn and dusk, with a maximum of 7 NR iterations required at dusk. Calling the QN pseudo-iteration on the first iteration (QN1, QN1-2, QN1-3 and QN1+; Figure 4b-e) reduces the number of NR iterations required to reach a stable solution on most timesteps, but increases the number of NR iterations at dawn on most days, therefore increasing the computational costs at these timesteps compared to the CNTL run. The experiments with the QN method first called on the second iteration (QN2, QN2-3 and QN2+; Figure 4f-h) consistently reduce the number of NR iterations required to reach a stable solution. Experiment QN2-3 is the most efficient of the three, reducing the number of NR iterations required ~~on the first timestep to 11~~, at the dusk timesteps to 6, and to 3.52 on average, giving a net average of 3.89 NR-equivalent iterations counting each QN psuedo-iteration as 27% of a full NR iteration. Experiment QN2+ shows diminishing returns compared to QN2-3, calling more QN pseudo-iterations for no reduction in NR iterations on most timesteps~~, although it does reduce the number of NR iterations on the first timestep to 8~~. The experiment with QN called on the third iteration only (QN3; Figure 3i) shows only marginal improvement compared to the CNTL scenario. Overall, QN2-3 most consistently reduces the net iteration count on average ~~and~~ at dawn and dusk in the Urban testcase. In some of the other scenarios, QN1-3 performed most efficiently (see supplement). However, in the urban scenario the QN2-3 experiment performs better at the dawn timesteps, when the QN1-3 experiment performs worse than the CNTL run. QN1-3 therefore shows signs of reduced robustness during the periods which are most challenging to solve, meaning it is unlikely to be able to handle the wide range of chemical states that will be simulated in the 3D model runs. We therefore use the QN2-3 setup for the 3D model runs, as UKCA_BOX results suggest it shows the most consistent improvements over the CNTL scenario.


(a) Urban, CNTL
1st = 12NR+0QN ~ 12.0NR
Max = 7NR+0QN ~ 7.00NR
Avg = 4.36NR+0.00QN
Net_Avg ~ 4.36±0.99NR
(b) Urban, QN 1
1st = 12NR+1QN ~ 12.27NR
Max = 7NR+1QN ~ 7.27NR
Avg = 3.62NR+1.00QN
Net_Avg ~ 3.89±1.36NR
(c) Urban, QN 1-2
1st = 11NR+2QN ~ 11.54NR
Max = 8NR+2QN ~ 8.54NR
Avg = 3.52NR+2.00QN
Net_Avg ~ 4.06±1.33NR
(d) Urban, QN 1-3
1st = 11NR+3QN ~ 11.81NR
Max = 7NR+3QN ~ 7.81NR
Avg = 3.45NR+2.18QN
Net_Avg ~ 4.04±1.27NR
(e) Urban, QN 1+
1st = 7NR+6QN ~ 8.62NR
Max = 6NR+5QN ~ 7.35NR
Avg = 3.34NR+2.34QN
Net_Avg ~ 3.97±1.03NR
(f) Urban, QN 2
1st = 12NR+1QN ~ 12.27NR
Max = 7NR+1QN ~ 7.27NR
Avg = 3.66NR+1.00QN
Net_Avg ~ 3.93±1.26NR
(g) Urban, QN 2-3
1st = 11NR+2QN ~ 11.54NR
Max = 6NR+2QN ~ 6.54NR
Avg = 3.52NR+1.34QN
Net_Avg ~ 3.89±1.13NR
(h) Urban, QN 2+
1st = 8NR+6QN ~ 9.62NR
Max = 6NR+4QN ~ 7.08NR
Avg = 3.49NR+1.49QN
Net_Avg ~ 3.89±1.09NR
(i) Urban, QN 3
1st = 12NR+1QN ~ 12.27NR
Max = 7NR+1QN ~ 7.27NR
Avg = 4.28NR+1.00QN
Net_Avg ~ 4.55±0.97NR
No. Iterations to solve
Timestep (hours)
NR Iterations
QN Iterations

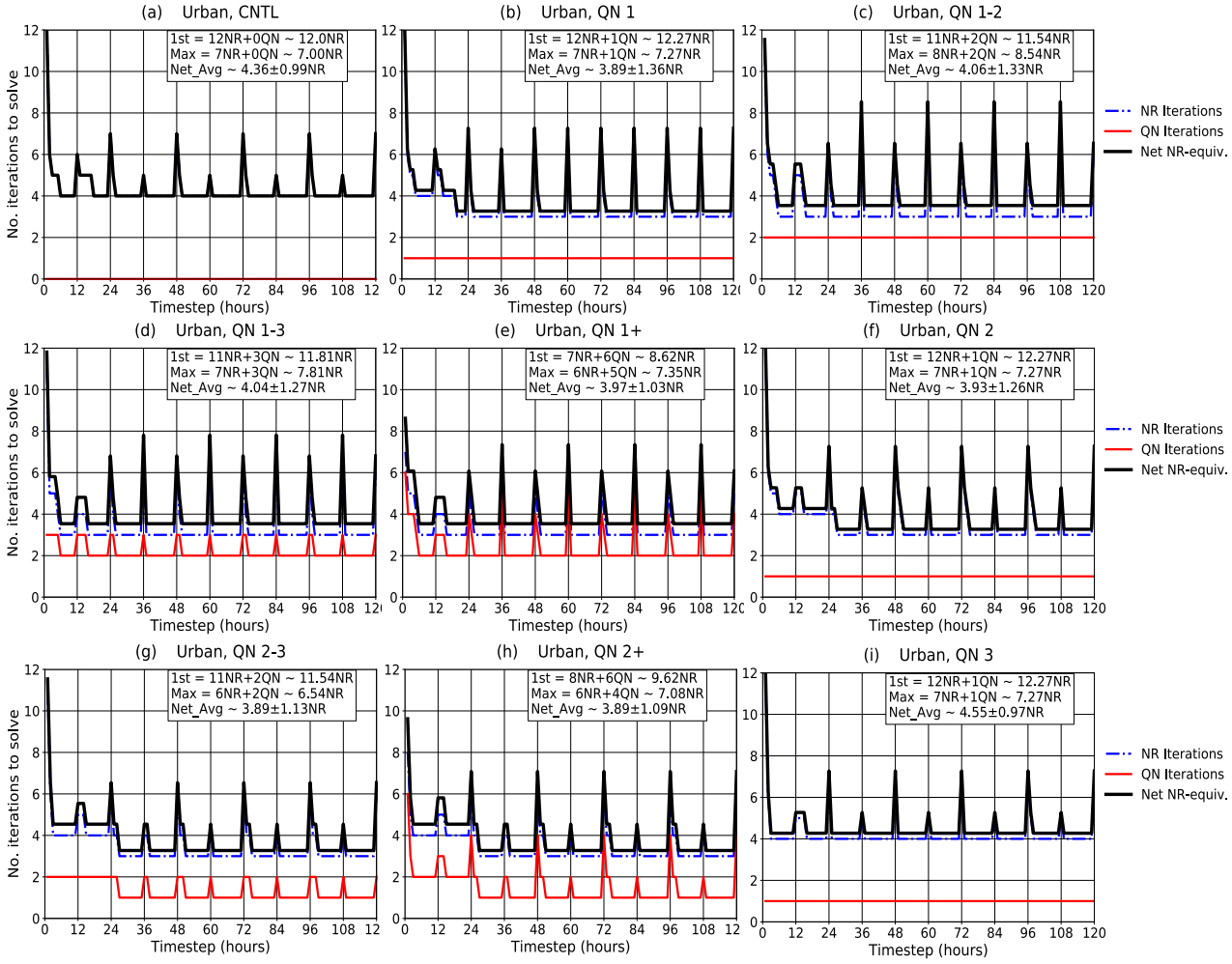


Figure 4. Plots of solver iteration (convergence) numbers for the original full Newton-Raphson (NR) method and Quasi-Newton (QN) methods, with QN pseudo-iterations only called on particular iteration(s). The CNTL scenario (top-left) only solves with NR iterations, and is equivalent to the solver in the release version of UKCA. The other 8 panels call QN pseudo-iterations on one or more iterations at each timestep. The blue dashed~~black~~ lines show number of NR iterations required to converge on a stable solution, while the red line shows number of QN pseudo-iterations required, and the black line total net number of NR-equivalent iterations to solved, calculated as NR + 0.27QN~~.~~ The white bands show periods with photolysis on, and grey band periods with photolysis off. The text in each panel gives the number of NR and QN iterations required to converge on the first timestep, the most difficult timestep after the first, and on average across the whole period~~, along with an estimate of the net computational time~~ in NR-equivalent iterations ~~based on the assumption that one QN pseudo-iteration takes 27% of the time of a full NR iteration~~.

3.2 UM-UKCA Simulations

In this section, we report our results for the full 3D global UM-UKCA simulations with the QN method implemented (on the original ASAD solver code) and without (classical NR method). We discuss these results from the perspectives of model performance (computational savings and stability) and prognostic evaluations (comparison of model physical values). All simulations were performed using version 10.6.1 of the model, applying the GA7.1 configuration at 1.875°×1.25° resolution with 85 vertical levels up to 85km (*N96L85*). Emissions were year 2000 CMIP5 emissions for all runs (Lamarque et. al., 2013). Aerosols were provided via a climatology. The UM-UKCA is a non-hydrostatic model which uses a regular longitude-latitude grid and a vertical hybrid height coordinate.

We have performed three sets of numerical experiments with two slightly different configurations of UKCA. The first version (StratTrop) uses the stratosphere-troposphere chemistry where all radiative feedback from UKCA trace gases was turned off and aerosol climatologies were used. This set-up allows for changing the chemical species whilst maintaining the same wind fields between the simulations. The UM-UKCA is parallelised by breaking the domain up into a chess-board pattern of sub-domains, defined by the number of processes given for the East-West (EW) and North-South (NS) directions. The solver iterates across all grid cells in the sub-domain until all have reached a stable solution. Thus, the computational speed is limited by the hardest-to-converge ("stiffest") grid cell in each subdomain. This configuration was run for 20 model years using 432-cores (24EW×18NS) in both control (CNTL) and quasi-Newton configurations (QN2-3). Additionally, four 1-year simulations were performed with additional timer diagnostics included using the Dr Hook package (ECMWF), two using 432-cores and two using 216-cores (18EW×12NS). In all these sets of simulations the initial start file was the same and the wind fields bit-compared at the end of the simulation.

A second set of simulations was performed using the stratosphere-troposphere chemistry combined with the GLOMAP-mode aerosol scheme (StratTrop+GLOMAP). This requires additional chemical species and reactions to be included on top of the standard StratTrop chemistry. In these simulations both CNTL and QN2-3 simulations were performed on 432-cores (24EW×18NS) for 20 model years (equivalent to the StratTrop simulations). However here both aerosols (via the direct and first and second indirect effects) and ozone, methane and nitrous oxide were coupled interactively to the MetUM dynamics via the model radiation scheme. This means that the wind fields ~~did not bit compare between~~ in these simulations were not identical as the small concentration~~s~~ changes introduced by the QN method resulted in global changes to the dynamical fields. Additionally, two 1-year simulations with timer diagnostics were also completed for CNTL and QN2-3 configurations.

3.2.1 Model Performance

We begin our discussion with an overview of the timing for each simulation set. These total time measurements are complemented by a robustness assessment, checking the number of times that iteration steps of the main chemistry solver are halved in order to reach the prescribed accuracy (that is, where UKCA spends more CPU in regions of stiff chemistry). This

initial analysis is then expanded to a more detailed analysis via time measurement maps of the simulations and iteration maps of the chemistry solver.

Table 4 gives the total wall-clock time measurement results for the four 20-year sets of simulations (jobs). A plot of the speed up for absolute wall-clock time is also included in the supplement (Figure S7). Using our suggested modification of the current algorithm leads to a net savings of ~2-3% over the full UM simulation despite the fact that the chemistry routine takes a relatively small part (5-10%) of the entire simulation (depending on the configuration). This suggests that using a (mixed) Quasi-Newton method has the potential to reduce the computational costs of other non-spatial systems with more intensive chemistry or even spatial systems modelled by partial differential equations that involves construction of a Jacobian for the computation of solutions. For the comparison of core components of the UKCA routines, we conducted 1-year long timer diagnostics analysis with the Dr Hook package. The results are tabulated in Table 5. It is found that the QN scheme speeds up the chemistry component between 12.7 % and13.5 % depending on the configuration.

Table 4. Computational speed-up using the QN method in comparison to regular Newton-Raphson method.

Chemistry	Number of Cores	Simulation	Mean wall-clock time for one month (s) ± 2× standard error	Speed-up (%)
StratTrop	432	CNTL	3525.7 ± 10.6	2.31 ± 0.01
		QN2-3	3444.2 ± 9.5	
StratTrop+GLOMAP	432	CNTL	4805.7 ± 20.6	2.93 ± 0.02
		QN2-3	4664.7 ± 14.2	

~~**Table 5. Average relative timings of the routines in the UM-UKCA with and without the QN method (1-year jobs each).**~~

~~Chemistry~~	~~Number of Cores~~	~~Simulation~~	~~Chemistry %~~	~~Convection %~~	~~Photolysis %~~	~~Chemistry speed-up %~~
~~StratTrop~~	~~432~~	~~CNTL~~	~~8.65~~	~~3.75~~	~~6.21~~	~~13.00~~
		~~QN2-3~~	~~7.70~~	~~3.83~~	~~6.36~~	
~~StratTrop+GLOMAP~~	~~432~~	~~CNTL~~	~~6.57~~	~~3.29~~	~~4.23~~	~~12.47~~
		~~QN2-3~~	~~5.84~~	~~3.35~~	~~4.29~~	
~~StratTrop~~	~~216~~	~~CNTL~~	~~11.0~~	~~4.42~~	~~7.37~~	~~13.48~~
		~~QN2-3~~	~~9.89~~	~~4.57~~	~~7.64~~	

Table 5. Average wallclock time in seconds (± 2 standard error) across all processors used for various UM components comparing the CNTL and QN2-3 methods. All are from 1-year simulations performed on a Cray XC40.

Chemistry	**StratTrop**		**StratTrop+GLOMAP**		**StratTrop**	
Cores	**432**		**432**		**216**	
Simulation	**CNTL**	**QN2-3**	**CNTL**	**QN2-3**	**CNTL**	**QN2-3**
Dynamics	12123 ± 22	12099 ± 23	15117 ± 28	15297 ± 27	18881 ± 27	18743 ± 30
Chemistry	4228 ± 26	3678 ± 16	4725 ± 28	4123 ± 19	9102 ± 96	7875 ± 75
Diagnostics	2951 ± 1	2979 ± 1	3628 ± 1	3641 ± 1	3098 ± 1	3108 ± 1
Photolysis	3038 ± 7	3038 ± 7	3041 ± 7	3030 ± 7	6082 ± 43	6084 ± 43
Convection	1833 ± 51	1828 ± 51	2367 ± 62	2366 ± 62	3648 ± 148	3637 ± 148
Radiation	1184 ± 10	1184 ± 10	1140 ± 10	1136 ± 9	2487 ± 34	2485 ± 34
UM Total	48871 ± 0	47730 ± 0	71900 ± 0	70596 ± 0	82561 ± 0	79600 ± 0
Chemistry Speed-up (%)	13.00		12.74		13.48	
UM Speed-up (%)	2.33		1.81		3.59	

A legitimate question is to check how Quasi-Newton methods, which are essentially based on approximations, change the robustness of the numerical scheme. This is particularly important since the modelled systems are generally under stiff conditions which are prone to instability. A poorly designed approximate method could wash out important information on the direction of the chemical evolution and cause the program to crash after some number of steps. To demonstrate that the approximation scheme that we propose is safe, we show in Table 6 the number of times the UKCA model halves the timestep (a sign that the chemical conditions at that particular location and time are such that the solution to fails to converge, oscillate, or even diverge, and therefore the timestep has to be reduced). According to Table 6, with the QN modification, the occurrence of halving the timestep is nearly two times less frequent compared to the original algorithm, suggesting that the mixed QN method can be more robust in chemically stiff environments, saving more computational time overall as halving the timestep significantly increases computational costs. The parallelisation of the UM-UKCA is such that the whole model can be held up by the few grid cells which fail to converge under the normal timestep. So improving the robustness of the solver potentially has much greater benefits to net computational efficiency than just the direct reduction in cost to solve the individual grid cells.

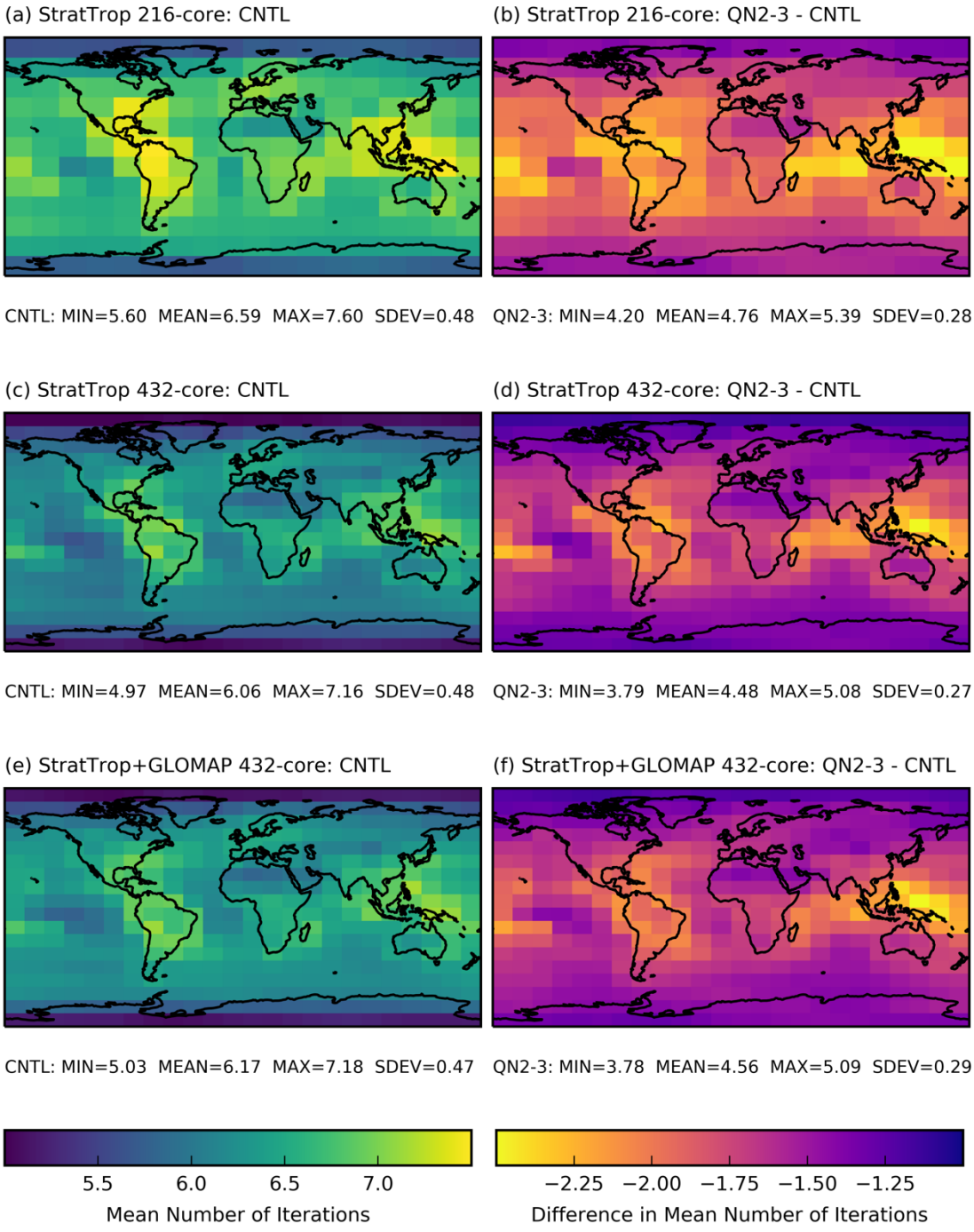
Table 6. Number of times that the solver needed to halve the timestep in order to avoid divergences or wild oscillations over one year of integration.

Chemistry	Number of Cores	Simulation	Number of halving steps	Fraction of total number of solver calls
StratTrop	216	CNTL	457344	0.00288
		QN2-3	270101	0.00170
	432	CNTL	436048	0.00137
		QN2-3	256019	0.00081
StratTrop+GLOMAP	432	CNTL	544532	0.00172
		QN2-3	328836	0.00104

Next, we make a grid point analysis of NR iterations to understand the origin of computational savings. In general, the time that it takes the solver to calculate final chemical concentrations on a grid point depends heavily on the ambient photo-chemical conditions at that point and time. So, the number of iterations in which the program exits the solver loop varies significantly across the domain.

Figure 5 shows maps of the mean number of iterations to convergence (averaged over column and time) for the 1-year simulations (one chemical timestep is equal to 1 model-hour) with the StratTrop (216 and 432 cores) and GLOMAP (432 cores) schemes. The CNTL simulations (left-hand column) clearly show regions where more iterations are required. The right-hand column shows the difference in mean number of iterations to convergence when using the QN2-3 method. Not only is

the mean number of NR iterations ~~is~~ reduced globally, but greater benefit is seen in the hot-spot regions noted in the CNTL



simulations.

Figure 5: ~~:~~ Left column (a,c,e): Maps of average NR iteration numbers for the 3 different 1-year standard UM Newton-Raphson solver (CNTL) simulations from Table 5, right column (b,d,f): differences between the quasi-Newton solver (QN2-3) and the

equivalent control simulation. The top plots (a,b) are StratTrop 216-core (18EW×12NS), the middle plots (c,d) are StratTrop 432-core (24EW×18NS), and the bottom plots (e,f) are StratTrop+GLOMAP 432-core. The quoted statistics are for the simulations and not for the differences.

By summing the total number of points through the 1-year year period according to number of iterations, a histogram of iteration numbers is produced which neatly summarizes performance of both methods (the CNTL and the QN cases). Figure 6 shows the histogram of the iteration numbers over all grid points for the 1-year simulations with the same StratTrop (216 and 432 cores) and GLOMAP (432 cores) schemes. The QN method greatly reduces the peak at 8 iterations, and allows the majority of solutions (approximately 70%) to be found in 4 or less NR iterations.

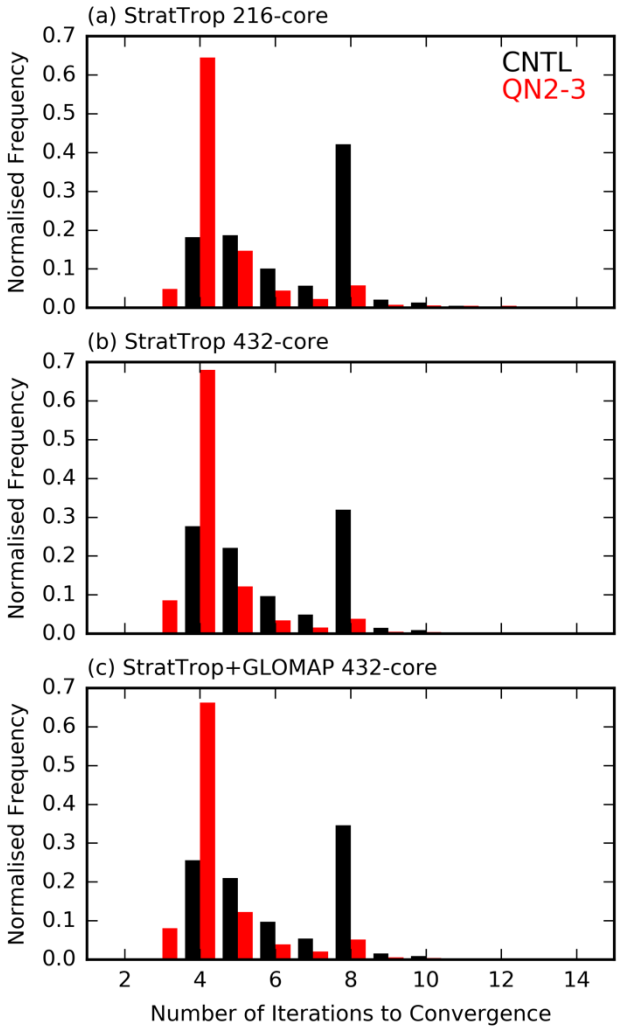


Figure 6: Histograms of the number of NR iterations to convergence for the StratTrop 216-core (top), StratTrop 432-core (middle), and StratTrop+GLOMAP 432-core (bottom) 1-year long simulations.

3.2.2 Model Evaluation

In this section we evaluate the accuracy of our proposed method. Recall from Section 3.1 that the QN method produces physical values which are very close to what the original method calculates even for fast changing species.

We test the accuracy of the two methods by comparing the model predictions for two different species which have very different lifetimes (O_3 and OH) and are key species that chemistry climate models need to simulate accurately (Monks et al., 2015). If the 3D model predictions for the two species which are on the opposite sides of the lifetime spectrum are very close, then it is very likely that physical values for all other species which have intermediate lifetimes will also be close.

For comparison of differences in values we consider only the StratTrop scenario in which ozone and other chemical feedbacks are not included. This avoids intrinsic perturbations dominating the solutions over long periods of time and ensures that the dynamics are identical between both simulations.

From the last 10-year average of two 20-year experiments StratTrop-CNTL and StratTrop-QN2-3, we see that O_3 concentrations (here plotted as a 10-year mean for the representative month of July) for the two experiments are very similar, as seen in Figure 7 (for zonal-mean differences on the left column and for surface differences on the right column). The same figures also show that the relative percentage differences (bottom row) between the two runs are negligible, being of the order of 0.01 % or smaller.

For the comparison of OH concentrations in the 20-year StratTrop-CNTL and StratTrop-QN2-3 experiments, Figure 8 shows the zonal-mean differences and surface value differences in the month of July. The difference values are slightly larger, but still only of the order of 0.1 % or smaller. Note that the largest percentage differences are seen in the areas with the smallest absolute OH concentrations. Almost everywhere else the fractional difference in OH is less than the tolerance of the solver (10^{-4}). It is also clear from the surface OH plots (Figure 8 b, d, f) that the differences in OH are so small that they are approaching the limits of the numerical scheme, as the sub-domains solved by each processor are clearly visible (being 24 in the X directory and 18 in the Y direction). This artefact appears because all grid cells in each sub-domain are iterated in the solver until all have converged. and so can introduce small numerical differences.

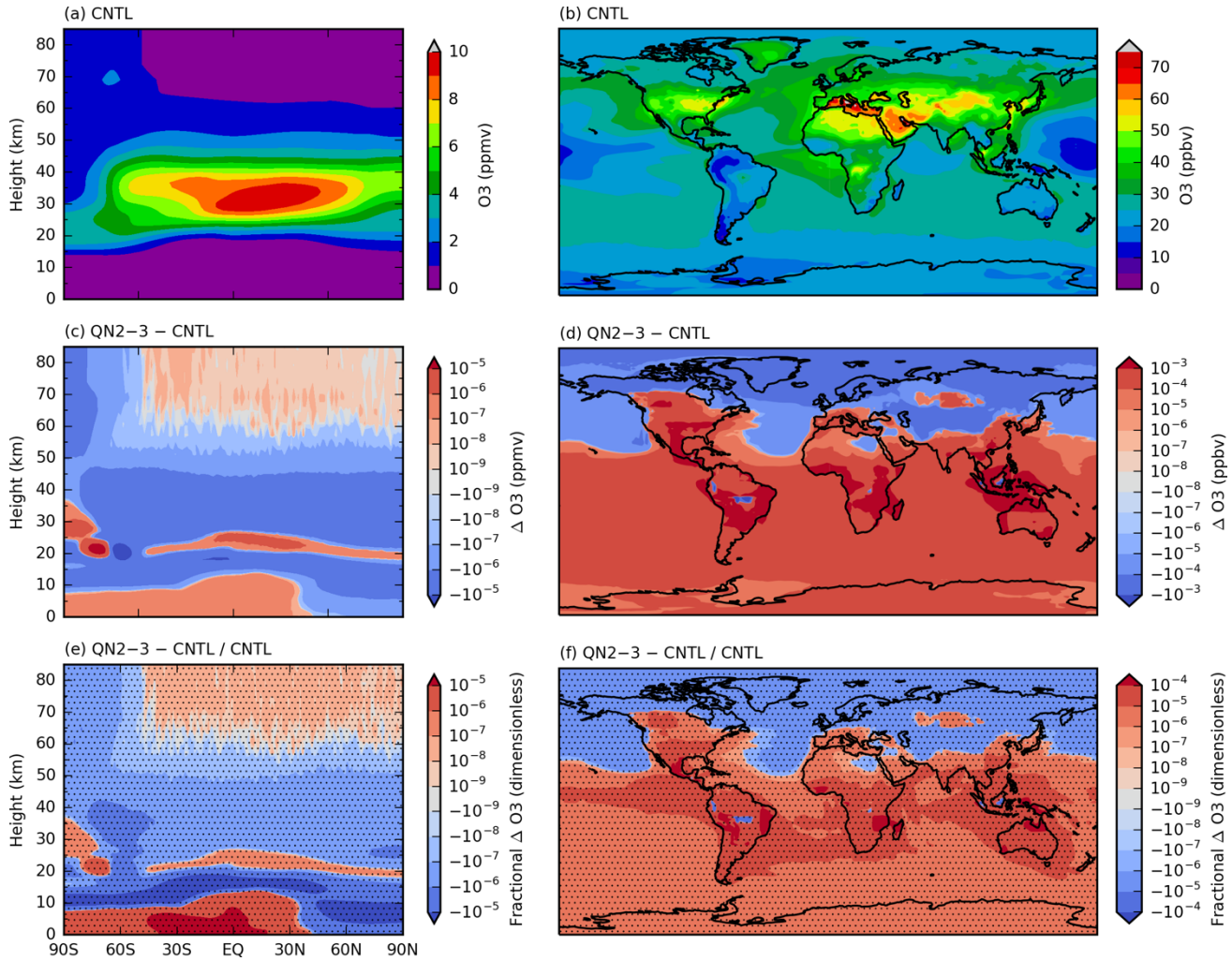


Figure 7: (a/c/e) Zonal-mean ozone from the last 10-years of the 20-year StratTrop 432-core simulations and (b/d/f) surface ozone from the last 10-years of the 20-year StratTrop 432-core simulations. (a/b) Ozone from the CNTL simulation. (c/d) Absolute differences QN2-3 simulation and the CNTL simulation. (e/f) Fractional differences between QN2-3 simulation and the CNTL simulation. Stippling in the (e) & (f) plots indicates that the values are below the convergence criterion of the chemical solver (10^{-4}).

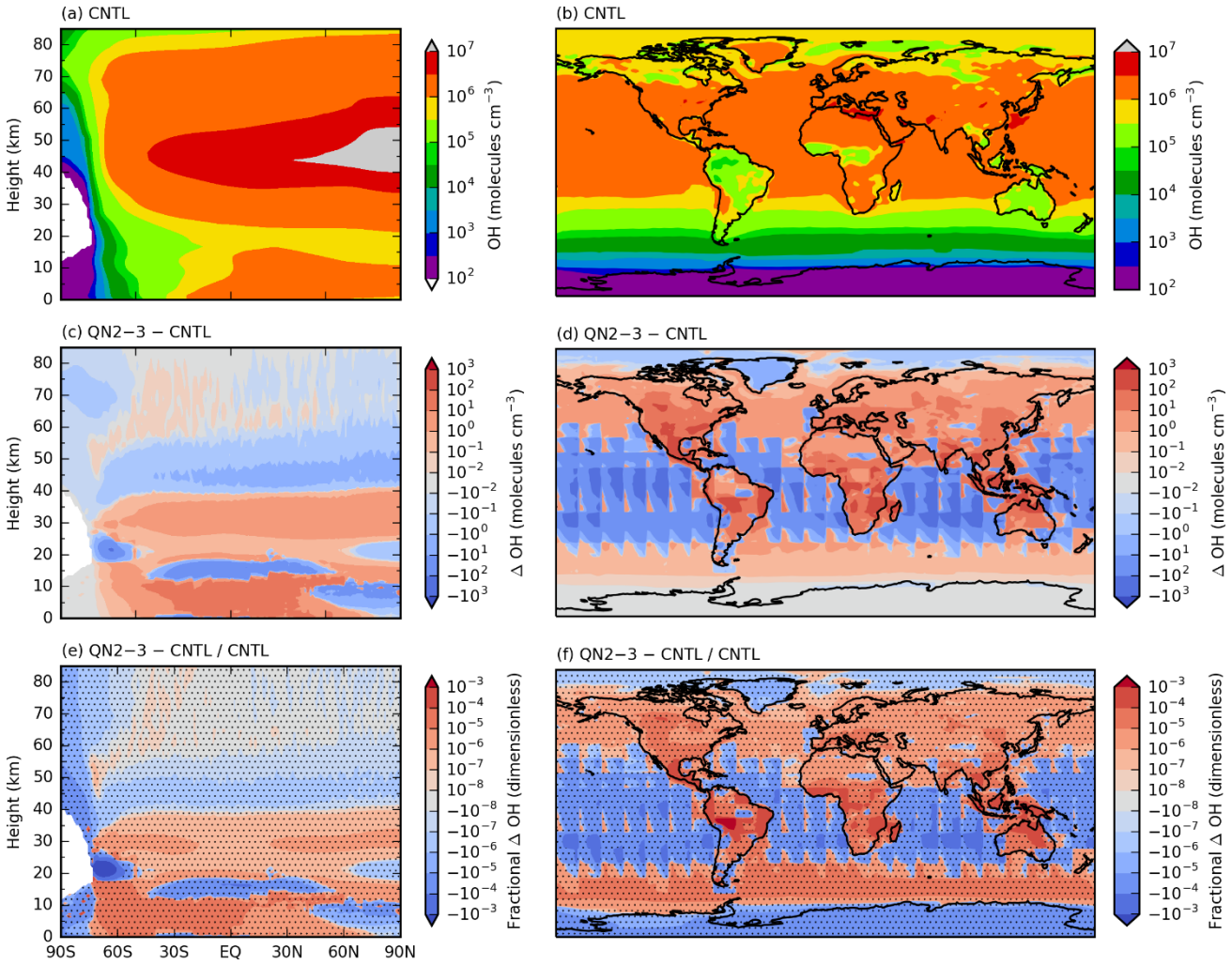


Figure 8: As Figure 7 but for OH. Note the use of a log-scale in the top (CNTL) plot. Note that the model domains are visible due to the extremely small differences in OH.

3.2.3 Analysis of the differences between simulations with UM-UKCA

In this subsection we give a quantitative analysis of the differences in the physical values obtained from the computations. In the strict sense of the word, there is actually no extra "error" associated with our proposed method of computation as both the classical NR and QN approaches give approximate solutions of the real DE within a chosen error tolerance (which is met by each method). Nevertheless, for completeness and comparison we will regard the NR computations (CNTL runs) as the "true" values and measure the difference in OH and O_3 fields ~~of~~ for the two runs using two different metrics defined below.

The figures in the previous sections provide maps of absolute and relative differences. Depending on the location of the point, these differences vary but always stay very small. In order to have a more quantitative measure of how different one particular run is from the other, we need a metric that will take into account all of the grid points and the corresponding errors. Considering the extreme low values of OH in certain regions, the most suitable metrics (Yu et. al., 2006) are the normalised mean absolute difference (NMAD) and normalised root mean square difference (NRMSD) which are respectively defined by

$$NMAD_S = \frac{\sum_i |c_{S,nr}{}^i(T) - c_{S,qn}{}^i(T)|}{\sum_i |c_{S,nr}{}^i(T)|} \quad (11),$$

$$\mathrm{NRMSD}_S = \sqrt{\frac{\sum_i |c_{S,nr}{}^i(T) - c_{S,qn}{}^i(T)|^2}{\sum_i |c_{S,nr}{}^i(T))|^2}} \quad (12).$$

where *S* denotes the species and *T* denotes the time at the end of the run. To measure the bias we calculate normalised mean bias which is defined as

$$NMB_S = \frac{\sum_i c_{S,nr}{}^i(T) - c_{S,qn}{}^i(T)}{\sum_i |c_{S,nr}{}^i(T)|} \quad (13).$$

Table 7 below shows the NMAD, NRMSD and NMB for the OH and O_3 species. A complete table showing NMAD, NRMSD and NMB for all species is provided in the supplement.

Table 7. Comparison of Newton-Raphson Vs Quasi-Newton methods by the metrics NMAD and NMRSD.

Chemistry	Species	Comparison	NMAD	NMRSD	NMB
StratTrop (432 cores)	OH	CNTL vs QN2-3	$3.6986\ 10^{-8}$	$3.6019\ 10^{-8}$	$3.0382\ 10^{-9}$
StratTrop (432 cores)	Ozone	CNTL vs QN2-3	$8.8374\ 10^{-7}$	$8.9908\ 10^{-7}$	$7.3761\ 10^{-7}$

We also plot the NMAD, NRMSD and NMB as a function of time (each month) in the last 10-year period for OH (Figures S8, S9 and S10 of the supplement respectively) and for O_3 (Figures S11, S12 and S13 of the supplement respectively). We observe that the differences are extremely small and stay bounded in time and do not grow, which indicates that the two methods reproduce essentially the same evolution. We remark that NMB values are smaller than NMAD in magnitude and do not grow in time, as expected. Similar conclusions can be drawn for the other species as shown in the supplement.

4. Conclusion

Atmospheric chemistry simulations are at the heart of coupled chemistry-climate models. Solving the complex sets of equations that represent the evolution of species comes at a high computational cost. In this article, we introduced a version of the Quasi-Newton method into the UKCA coupled climate model. The Quasi-Newton method demonstrates improvements, in multiple ways, over the classical Newton-Raphson method used in the UKCA model chemistry solver.

The main benefit of the QN approach, as discussed in Section 3, is its ability to reduce the computational time for the simulations. The advantages, however, are not limited to reducing the costs of chemistry calculations. The computations are more robust against stiff chemical environments thereby reducing the possibility of divergence and instability in computations. On parallel platforms, even when there is no danger of instability, robustness actually can translate into extra computational gain as the method save further time by avoiding unnecessary wait times in the sub-domains. Overall, we see a reduction in total computational costs of the whole UKCA model of approximately 3%, corresponding to a reduction of approximately 15% in the chemistry routines. Whilst this may not seem like a big reduction, it is significant given the high costs associated with the rest of the coupled UKCA model. In practice, a 3% reduction of costs for a large study involving 10,000 model years corresponds to 300 model years saved, roughly 100 real days of supercomputer-time with the current setup.

We also demonstrated that~~,~~ the suggested method, while improving the performance, does not deteriorate the accuracy of physical predictions, which is an obvious requirement for any proposed method. From the cross comparisons under different computational environments (UKCA_BOX or parallel UM simulations), different chemical scenarios (interactive or noninteractive) for a large spectrum of chemical species (varying from very long lifetime or short lifetime) the method maintains the same level accuracy with the original method.

Another feature of our approach is its flexibility for use with many existing chemistry solving. Whilst this work focussed specifically on the UKCA, the algorithm can be easily integrated to the existing codes of the other (unrelated) coupled chemical system solvers. If implemented in a chemical transport model, for example, one would expect the overall benefit to be greater, due to the greater proportion of computational expense of the chemical solver due to the lack of other online physical processes. As sketched in Section 2, it is also simple to detach the algorithm from the modified program and revert back to the original algorithm if desired using options defined in the namelist. Furthermore, since the method is quite generic, it can be used beyond solving chemical systems. We think that it will be just as easy to implement the method to other components of the climate model. For instance, solving systems of time dependent nonlinear (partial) differential equations which can be cast into a problem of solving systems of nonlinear algebraic equations at each timestep.

Finally, we remark that we have focused on one particular Quasi-Newton approach which took advantage of available information and use the to replace costly Jacobian construction and linear system solving routines which proved to work robustly under fairly general conditions. There are also other Newton type methods that avoid or reduce Jacobian construction (Brown and Saad, 1990). Although these methods pursue relatively different strategies (and hence require more substantial changes to a classical NR type algorithm) it would be interesting to investigate their numerical capability.

54 Code Availability

Due to intellectual property right restrictions, we cannot provide either the source code or documentation papers for the UM or JULES. However, we provide a pseudo code for the NR + QN routine part of the DE system solver of the UKCA (see Appendix A below).

Obtaining the UM. The Met Office Unified Model is available for use under licence. The code is available in the UM trunk from version 10.8. Branches are also available at vn10.7 and vn10.6.1. A number of research organisations and national meteorological services use the UM in collaboration with the Met Office to undertake basic atmospheric process research, produce forecasts, develop the UM code and build and evaluate Earth system models. For further information on how to apply for a licence see http://www.metoffice.gov.uk/research/modelling-systems/unified-model.

Appendix A (Pseudo Code for NR+QN Routine)

! **Pseudo Code for Solving the Equation *F(c)=0***
! Inside the new chemistry step: determine the concentrations for the next step...

…
…
err = 10^{-4}
…

Update tendencies ($\boldsymbol{f}(\boldsymbol{c}_*)$) at the time of the current chemistry step (t_*)

Make an initial guess for the algebraic system as an input to the iterative solver
$\boldsymbol{c} = \boldsymbol{c}_* + \boldsymbol{f}(\boldsymbol{c}_*)\ del\_t$

! Main NR Iteration loop starts

! Iteration counter: k, maximum iteration counter: max_iter

Do k=1,max_iter

! Update the ***F*** vector and store it
$\boldsymbol{F} = \frac{\boldsymbol{c}-\boldsymbol{c}_*}{del\_t} - \boldsymbol{f}$
Fold =F

! Jacobian construction and linear system solving
Compute exact Jacobian $\mathbf{J}(\boldsymbol{c})$ of the ***F*** vector()

Solve for the new increment ***del_c*** in the equation
$\mathbf{J}(\boldsymbol{del\_c}) = \boldsymbol{F}$

$err\_c = \frac{\text{maxval}(abs(\boldsymbol{delt\_c}))}{maxval(abs(\boldsymbol{c}))}$

! Updating the $\boldsymbol{c}$ values
Perform treatments for troublesome convergence (e.g. β dampening factor) or
Filtering of possible negative values in components of
$\boldsymbol{c} = \boldsymbol{c} + \beta\, \boldsymbol{del\_c}$

! Test and decide if QN step will be taken
! This can be done on iterations 2 ≤ k ≤ 50, and recommended on steps 2 & 3
! This step will not be done if the $\boldsymbol{c}$ vector converged and the routine is about to exit

If ($err\_c \geq err$ .AND. $choice\_qn$) Then

Update the tendencies

Update the $\boldsymbol{F}$ vector
$\boldsymbol{F} = \frac{\boldsymbol{c}-\boldsymbol{c}_*}{\Delta t} - \boldsymbol{f}$
$\boldsymbol{delF}=\boldsymbol{F}-\boldsymbol{Fold}$

! QN approximation below …

Compute the Jacobian modification factor
$a = \frac{DOT\_product(\boldsymbol{delF},\boldsymbol{F})}{DOT\_product(\boldsymbol{del\_c},\boldsymbol{del\_c})}$

Re-solve for the newer increment $\boldsymbol{del\_c}$
$\mathbf{J}\,(\boldsymbol{del\_c}) = (1-a)\boldsymbol{F}$

Update $\boldsymbol{c}$ values
$\boldsymbol{c} = \boldsymbol{c} + \beta\, \boldsymbol{del\_c}$

End If

$k=k+1$

End Do

Acknowledgement

We thank Oliver Wild for many useful discussions. The first author thanks Nigel Wood and Olaf Morgenstern for helpful comments. We also thank Alan Hewitt and Stuart Whitehouse for reviewing the code.

Model integrations have been performed using the ARCHER UK National Supercomputing Service and the MONSooN system, a collaborative facility supplied under the Joint Weather and Climate Research Programme, which is a strategic partnership between the UK Met Office and the Natural Environment Research Council. This work used the NEXCS HPC

facility provided by the Natural Environment Research Council. We thank NCAS for providing support for the UKCA model development.

The first and last authors were supported under the ERC (ACCI) grant (project number 267760). Alex T. Archibald and Scott Nicholls thank the Isaac Newton Trust under whose auspices this work was funded.

References

N. L. Abraham, A. T. Archibald, N. Bellouin, O. Boucher, P. Braesicke, A. Bushell, K. Carslaw, B. Collins, M. Dalvi, K. Emmerson, G. Folberth, J. Haywood, A. Hewitt, C. Johnson, Z. Kipling, H. Macintyre, G. Mann, P. Telford, J. Merikanto, O. Morgenstern, F. O'Connor, C. Ordonez, S. Osprey, K. Pringle, J. Pyle, J. Rae, C. Reddington, N. Savage, D. Spracklen, P. Stier, R. West, J. Mulcahy, S. Woodward, I. Boutle, M. T. Woodhouse, Unified Model Documentation Paper 084: United Kingdom Chemistry and Aerosol (UKCA) Technical Description Met UM Version 10.6, https://code.metoffice.gov.uk/doc/um/vn10.6/papers/umdp_084.pdf (for the version of the model used here)

Atkinson K., Introduction to Numerical Analysis, John Wiley & Sons Inc., 1989

Banerjee, A., Maycock, A. C., Archibald, A. T., Abraham, N. L., Telford, P., Braesicke, P., and Pyle, J. A.: Drivers of changes in stratospheric and tropospheric ozone between year 2000 and 2100, Atmos. Chem. Phys., 16, 2727-2746, 2016.

Brandt A, Multilevel adaptive solutions to boundary value problems. Math. Comp., (31), 333, 1977

Brown P. N., Saad Y., Hybrid Krylov methods for nonlinear system of equations, SIAM Journal of Science and Statistical Computations, 11, 450-481, 1990

Broyden, C. G., A Class of Methods for Solving Nonlinear Simultaneous Equations. Mathematics of Computation. American Mathematical Society. 19 (92): 577–593, 1965

Cariolle D., Moinat P., Teyssedre H., Giraud L., Josse B., Lefevre F., ASIS v1.0: An adaptive solver for the simulation of atmospheric chemistry, Geosci. Mod. Dev, Discuss, DOI:10.5194/gmd-2016-281, 2016

Chan T. F., Jackson K. R., Nonlinearly preconditioned Kylov subspace methods for discrete Newton algortihms, SIAM Journal of Science and Statistical Computation, 5, 533-542, 1984

GD Carver, PD Brown, and O Wild. The ASAD atmospheric chemistry integration package and chemical reaction database. Computer Physics Communications, 105(2-3):197–215, OCT 1997.

Damian, V., Sandu, A., Damian, M., Potra, F., & Carmichael, G. R. (2002). The kinetic preprocessor KPP - a software environment for solving chemical kinetics. Computers and Chemical Engineering, *26*, 1567–1579.

Davidon, W. C., Variable metric method for minimization, SIAM Journal on Optimization, 1: 1–17, 1991

ECMWF, Documentation, https://software.ecmwf.int/wiki/display/OIFS/Documentation.
The direct download link is at: https://software.ecmwf.int/wiki/download/attachments/19661682/drhook.pdf

Fletcher, R., A New Approach to Variable Metric Algorithms, *Computer Journal*, **13** (3): 317–322, 1970
Glotfelty T, He J, Zhang Y. Impact of future climate policy scenarios on air quality and aerosol-cloud interactions using an advanced version of CESM/CAM5: Part I. model evaluation for the current decadal simulations, Atmospheric Environment, 152: 222-239, 2017

Goldfarb, D., A Family of Variable Metric Updates Derived by Variational Means, Mathematics of Computation, **24** (109): 23–26, 1970

Heal MR, Heaviside C, Doherty RM, Vieno M, Stevenson DS, Vardoulakis S. Health burdens of surface ozone in the UK for a range of future scenarios, Environment International, 61: 36-44, 2013

Hewitt, H. T., Copsey, D., Culverwell, I. D., Harris, C. M., Hill, R. S. R., Keen, A. B., McLaren, A. J., and Hunke, E. C., Design and implementation of the infrastructure of HadGEM3: The next-generation Met Office climate modelling system. Geosci. Model Dev. 4, 223–253 (2011).

IPCC, 2013: Climate Change 2013: The Physical Science Basis. Contribution of Working Group I to the Fifth Assessment Report of the Intergovernmental Panel on Climate Change [Stocker, T.F., D. Qin, G.-K. Plattner, M. Tignor, S.K. Allen, J. Boschung, A. Nauels, Y. Xia, V. Bex and P.M. Midgley (eds.)]. Cambridge University Press, Cambridge, United Kingdom and New York, NY, USA, 1535 pp.

Knoll D. A., Keyes D. E., Newton-Krylov methods: A survey of approaches and applications, Journal of Computational Physics, 193, 357-397, 2004

Kvaalen, E., A faster Broyden method. BIT Numerical Mathematics. SIAM. 31 (2): 369–372, 1991

Kelley C. T., Iterative Methods for Linear and Nonliner Equations, SIAM, Philadelphia, 1995

Lamarque, J. F., Shindell, D. T., Josse, B., Young, P. J., Cionni, I., Eyring, V., Bergmann, D., Cameron-Smith, P., Collins, W. J., Doherty, R., Dalsoren, S., Faluvegi, G., Folberth, G., Ghan, S. J., Horowitz, L. W., Lee, Y. H., MacKenzie, I. A., Nagashima, T., Naik, V., Plummer, D., Righi, M., Rumbold, S. T., Schulz, M., Skeie, R. B., Stevenson, D. S., Strode, S., Sudo, K., Szopa, S., Voulgarakis, A., Zeng, G. (2013). The atmospheric chemistry and climate model intercomparison Project (ACCMIP): Overview and description of models, simulations and climate diagnostics. *Geoscientific Model Development*, *6*, 179–206. https://doi.org/10.5194/gmd-6-179-2013

Lauritzen P. H., Nair, R. D., Ullrich P., A conservative semi-Lagrangian multi-tracer transport scheme (CSLAM) on the cubed-sphere grid, Journal of Computational Physics, (229), 1401-1424, 2009

Lu T.~~ianfeng~~., Chung L., Yoo C. S., Chen J. H., Dynamic stiffness for direct numerical simulations, Combustion and Flame, (156), 1542-1551, 2009

Mann, G. W., Carslaw, K. S., Spracklen, D. V., Ridley, D. A., Manktelow, P. T., Chipperfield, M. P., Pickering, S. J., and Johnson, C. E.: Description and evaluation of GLOMAP-mode: a modal global aerosol microphysics model for the UKCA composition-climate model, Geosci. Model Dev., 3, 519-551, https://doi.org/10.5194/gmd-3-519-2010, 2010.

Mitsakou C., Helmis C., Housiadas C., Eulerian modelling of lung deposition with sectional representation of aerosol dynamics, Journal of Aerosol Science, (36), 75-94, 2005

Monks, P.S., Archibald, A.T., Colette, A., Cooper, O., Coyle, M., Derwent, R., Fowler, D., Granier, C., Law, K.S., Mills, G.E. and Stevenson, D.S., 2015. Tropospheric ozone and its precursors from the urban to the global scale from air quality to short-lived climate forcer. Atmospheric Chemistry and Physics, 15(15), pp.8889-8973.

Morgenstern, O., Braesicke, P., O'Connor, F. M., Bushell, A. C., Johnson, C. E., Osprey, S. M., and Pyle, J. A.: Evaluation of the new UKCA climate-composition model – Part 1: The stratosphere, Geosci. Model Dev., 2, 43-57, https://doi.org/10.5194/gmd-2-43-2009, 2009.

O'Connor, F. M., Johnson, C. E., Morgenstern, O., Abraham, N. L., Braesicke, P., Dalvi, M., Folberth, G. A., Sanderson, M. G., Telford, P. J., Voulgarakis, A., Young, P. J., Zeng, G., Collins, W. J., and Pyle, J. A.: Evaluation of the new UKCA climate-composition model – Part 2: The Troposphere, Geosci. Model Dev., 7, 41-91, https://doi.org/10.5194/gmd-7-41-2014, 2014.

Shanno, D. F.., Conditioning of quasi-Newton methods for function minimization, Mathematics of Computation, **24** (111): 647–656, 1970

Tilmes, S., Lamarque, J. F., Emmons, L. K., Kinnison, D. E., Ma, P. L., Liu, X., Ghan, S., Bardeen, C., Arnold, S., Deeter, M., Vitt, F., Ryerson, T., Elkins, J. W., Moore, F., Spackman, J. R., Val Martin, M., Description and evaluation of tropospheric chemistry and aerosols in the Community Earth System Model (CESM1.2). *Geoscientific Model Development*, *8*, 1395–1426. https://doi.org/10.5194/gmd-8-1395-2015, 2015.

Viallet, Maxime, Tom Goffrey, Isabelle Baraffe, Doris Folini, Chris Geroux, M. V. Popov, Jane Pratt, and Rolf Walder. "A Jacobian-free Newton-Krylov method for time-implicit multidimensional hydrodynamics-Physics-based preconditioning for sound waves and thermal diffusion." Astronomy & Astrophysics, 586 A153, (2016).

Walters, D., Boutle, I., Brooks, M., Melvin, T., Stratton, R., Vosper, S., Wells, H., Williams, K., Wood, N., Allen, T., Bushell, A., Copsey, D., Earnshaw, P., Edwards, J., Gross, M., Hardiman, S., Harris, C., Heming, J., Klingaman, N., Levine, R., Manners, J., Martin, G., Milton, S., Mittermaier, M., Morcrette, C., Riddick, T., Roberts, M., Sanchez, C., Selwood, P., Stirling, A., Smith, C., Suri, D., Tennant, W., Vidale, P. L., Wilkinson, J., Willett, M., Woolnough, S., and Xavier, P.: The Met Office Unified Model Global Atmosphere 6.0/6.1 and JULES Global Land 6.0/6.1 configurations, Geosci. Model Dev., 10, 1487-1520, https://doi.org/10.5194/gmd-10-1487-2017, 2017.

Ortega J., Rheinboldt W., Iterative solutions of Nonlinear Equations in several variables, Acadmic Press, Boston, 1970

Sandu A., Verwer J. G., Blom J. G., Spee E. J., Carmichael G. R., Potra F. A., Benchmarking stiff ode solvers for atmospheric chemistry problems II: Rosenbrock solvers, Atmospsheric Environment, (31), 3469-3472, 1997

Whitehouse L. E., Tomlin A. S., Pilling M. J., Systematic Reduction of complex tropospheric chemical mechanisms, Part I: sensitivity and time-scale analyses, Atmospheric Chemistry and Physics, 4, 202502056, 2004

Wild O., Prather M. J., Excitation of the primary tropospheric chemical mode in a global three dimensional model. Journal of Geophysical Research-Atmospheres, 105(D20):24647–24660, OCT 27 2000.

Wild, O., Zhu, X. I. N., & Prather, M. J., Fast-J: Accurate Simulation of In- and Below-Cloud Photolysis in Tropospheric Chemical Models. *Journal of Atmospheric Chemistry*, *37*, 245–282, 2000.

Yu S., Eder B., Dennis R., Chu S., Schwartz S. E., New unbiased symmetric metrics for evaluation of air quality models, Atmospheric Science Letters, 7, 26-34, 2006

SUPPLEMENTARY MATERIALS FOR THE ARTICLE

Quasi-Newton Methods for Atmospheric Chemistry Computations: Implementation in UKCA UM ~~Vn10~~vn10.8

Emre Esenturk[1,2], Nathan Luke Abraham[2,3], Scott Archer-Nicholls[2], Christina Mitsakou[2,b], Paul Griffiths[2,3], Alex Archibald[2,3], John Pyle[2,3]

[1]Mathematics Institute, University of Warwick, Coventry, CV4 7AL, UK
[2]Department of Chemistry, University of Cambridge, Cambridge, CB2 1EW, UK
[3]National Centre for Atmospheric Science, Cambridge, UK
[b] Currently at the Center for Radiation, Chemical Environments and Hazards, Public Health England, Chilton, Oxon, OX11 0RQ, UK

1. Extended UKCA Box-Model analysis.

The following is a short discussion of the extra model scenarios carried out for the ~~BoxModel~~ BOX-MODEL analysis: Rural, Marin and Strat. Equivalent figures to Figures 2 and 3 from the main paper are included for each of these scenarios. The Urban scenario proved to be the most challenging to solve, and have the largest errors, so is shown in the main paper. The QN methods proved to be robust and improved solving speed in all scenarios. However, the choice of what was the ideal iteration number to call the QN iterations varied from scenario to scenario. In the Urban scenario, calling the QN iteration after the first NR iteration often increased the number of iterations required to reach a stable solution, so was counter-productive. In the Rural, Marine and Strat runs, calling the QN method on the first iteration is generally effective at reducing the number of ~~FN~~ NR iterations required. However, given that the Urban cases were generally the most challenging to solve, the risk of calling the QN on the first iteration causing increase instability was considered to be great. In all scenarios, the QN2-3 experiment consistently improved over the CTNL run, so was the setup chosen to be used in the 3D model.

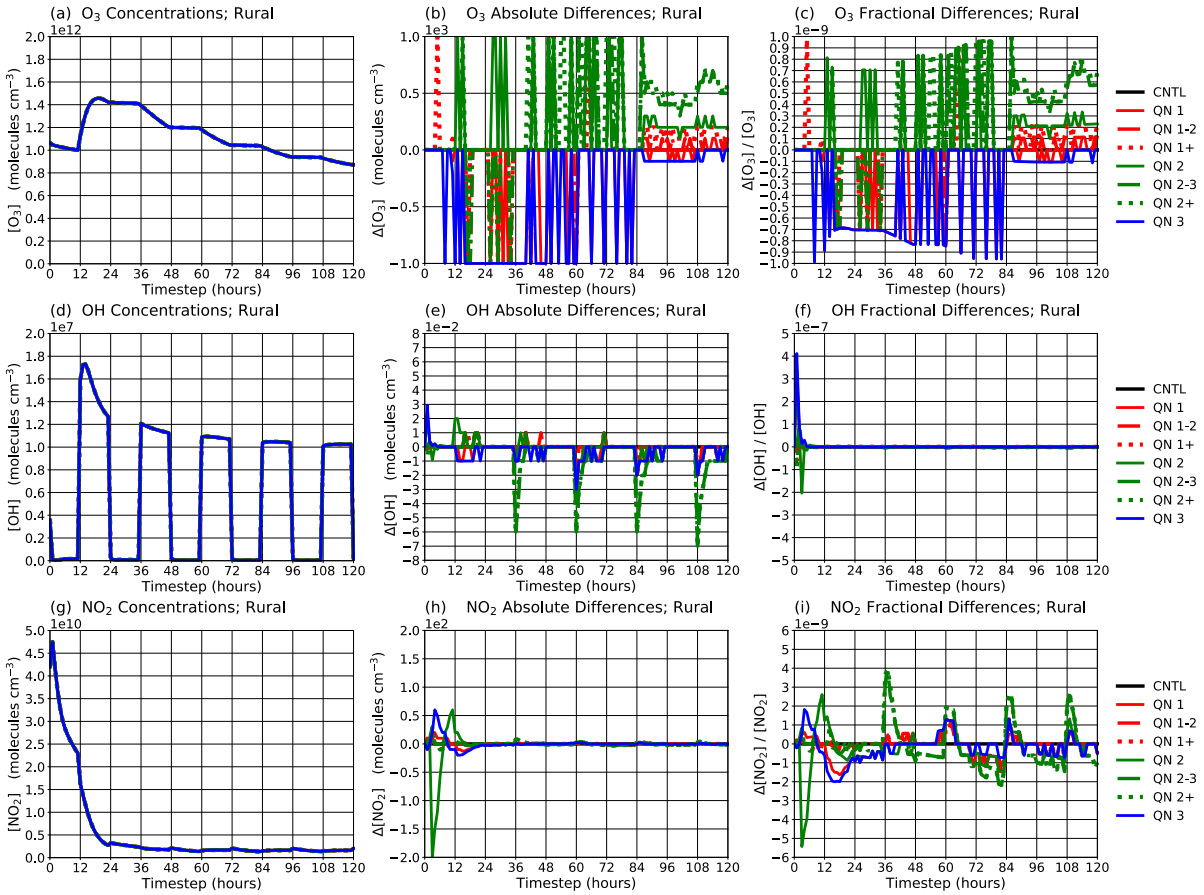
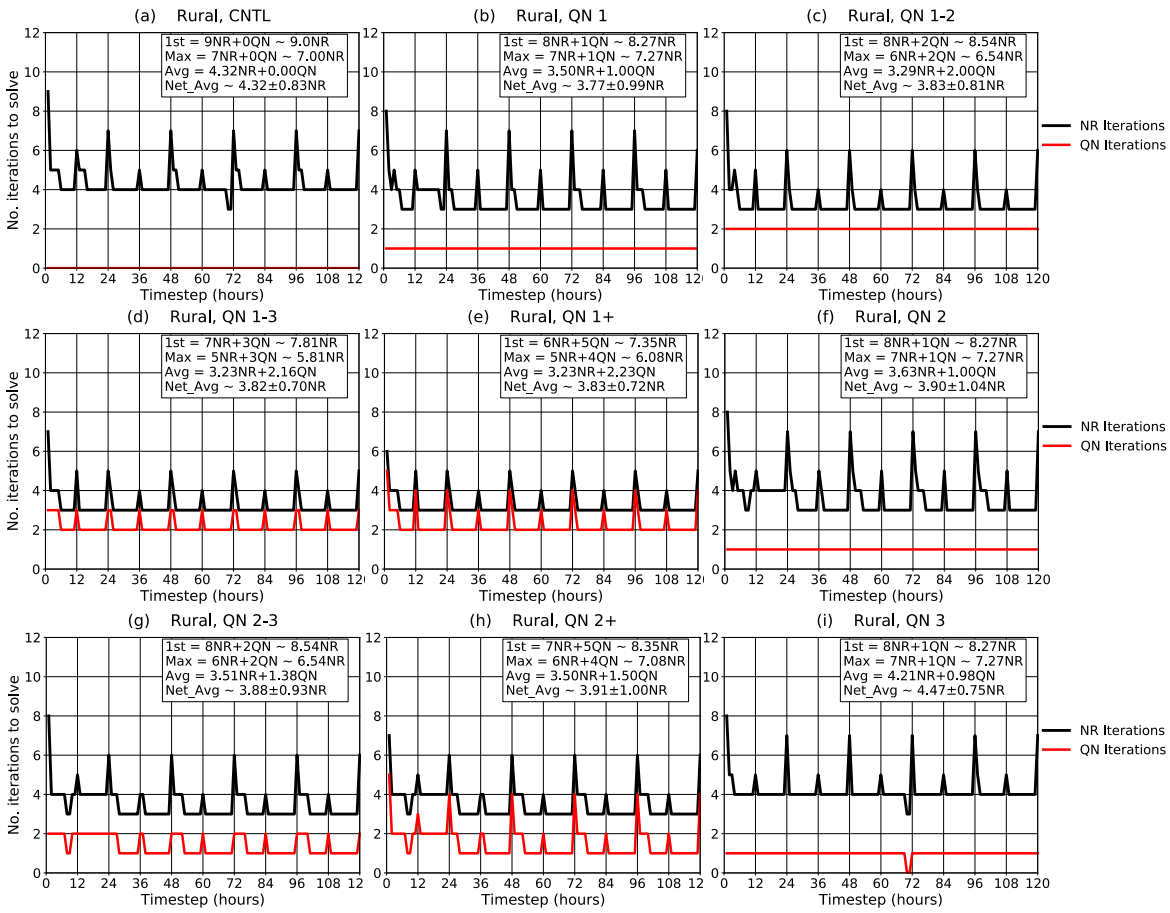


Figure S1. Same as Figure 2 in main paper, but for the Rural scenario


(a) Rural, CNTL
1st = 9NR+0QN ~ 9.0NR
Max = 7NR+0QN ~ 7.00NR
Avg = 4.32NR+0.00QN
Net_Avg ~ 4.32±0.83NR
(b) Rural, QN 1
1st = 8NR+1QN ~ 8.27NR
Max = 7NR+1QN ~ 7.27NR
Avg = 3.50NR+1.00QN
Net_Avg ~ 3.77±0.99NR
(c) Rural, QN 1-2
1st = 8NR+2QN ~ 8.54NR
Max = 6NR+2QN ~ 6.54NR
Avg = 3.29NR+2.00QN
Net_Avg ~ 3.83±0.81NR
(d) Rural, QN 1-3
1st = 7NR+3QN ~ 7.81NR
Max = 5NR+3QN ~ 5.81NR
Avg = 3.23NR+2.16QN
Net_Avg ~ 3.82±0.70NR
(e) Rural, QN 1+
1st = 6NR+5QN ~ 7.35NR
Max = 5NR+4QN ~ 6.08NR
Avg = 3.23NR+2.23QN
Net_Avg ~ 3.83±0.72NR
(f) Rural, QN 2
1st = 8NR+1QN ~ 8.27NR
Max = 7NR+1QN ~ 7.27NR
Avg = 3.63NR+1.00QN
Net_Avg ~ 3.90±1.04NR
(g) Rural, QN 2-3
1st = 8NR+2QN ~ 8.54NR
Max = 6NR+2QN ~ 6.54NR
Avg = 3.51NR+1.38QN
Net_Avg ~ 3.88±0.93NR
(h) Rural, QN 2+
1st = 7NR+5QN ~ 8.35NR
Max = 6NR+4QN ~ 7.08NR
Avg = 3.50NR+1.50QN
Net_Avg ~ 3.91±1.00NR
(i) Rural, QN 3
1st = 8NR+1QN ~ 8.27NR
Max = 7NR+1QN ~ 7.27NR
Avg = 4.21NR+0.98QN
Net_Avg ~ 4.47±0.75NR
No. iterations to solve
Timestep (hours)
NR Iterations
QN Iterations

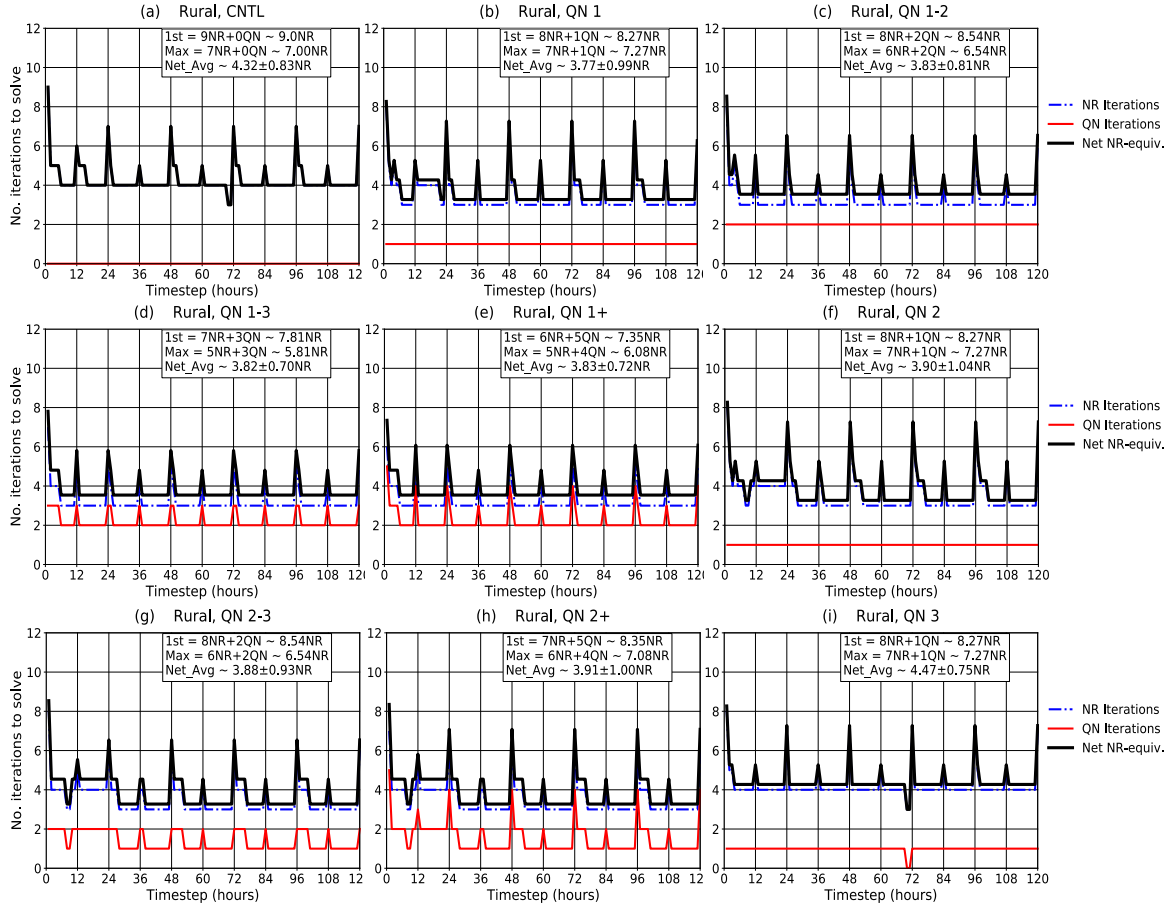

(a) Rural, CNTL
1st = 9NR+0QN ~ 9.0NR
Max = 7NR+0QN ~ 7.00NR
Net_Avg ~ 4.32±0.83NR
(b) Rural, QN 1
1st = 8NR+1QN ~ 8.27NR
Max = 7NR+1QN ~ 7.27NR
Net_Avg ~ 3.77±0.99NR
(c) Rural, QN 1-2
1st = 8NR+2QN ~ 8.54NR
Max = 6NR+2QN ~ 6.54NR
Net_Avg ~ 3.83±0.81NR
(d) Rural, QN 1-3
1st = 7NR+3QN ~ 7.81NR
Max = 5NR+3QN ~ 5.81NR
Net_Avg ~ 3.82±0.70NR
(e) Rural, QN 1+
1st = 6NR+5QN ~ 7.35NR
Max = 5NR+4QN ~ 6.08NR
Net_Avg ~ 3.83±0.72NR
(f) Rural, QN 2
1st = 8NR+1QN ~ 8.27NR
Max = 7NR+1QN ~ 7.27NR
Net_Avg ~ 3.90±1.04NR
(g) Rural, QN 2-3
1st = 8NR+2QN ~ 8.54NR
Max = 6NR+2QN ~ 6.54NR
Net_Avg ~ 3.88±0.93NR
(h) Rural, QN 2+
1st = 7NR+5QN ~ 8.35NR
Max = 6NR+4QN ~ 7.08NR
Net_Avg ~ 3.91±1.00NR
(i) Rural, QN 3
1st = 8NR+1QN ~ 8.27NR
Max = 7NR+1QN ~ 7.27NR
Net_Avg ~ 4.47±0.75NR
No. iterations to solve
Timestep (hours)
NR Iterations
QN Iterations
Net NR-equiv.

Figure S2. Same as Figure 3 in main paper, but for the Rural scenario

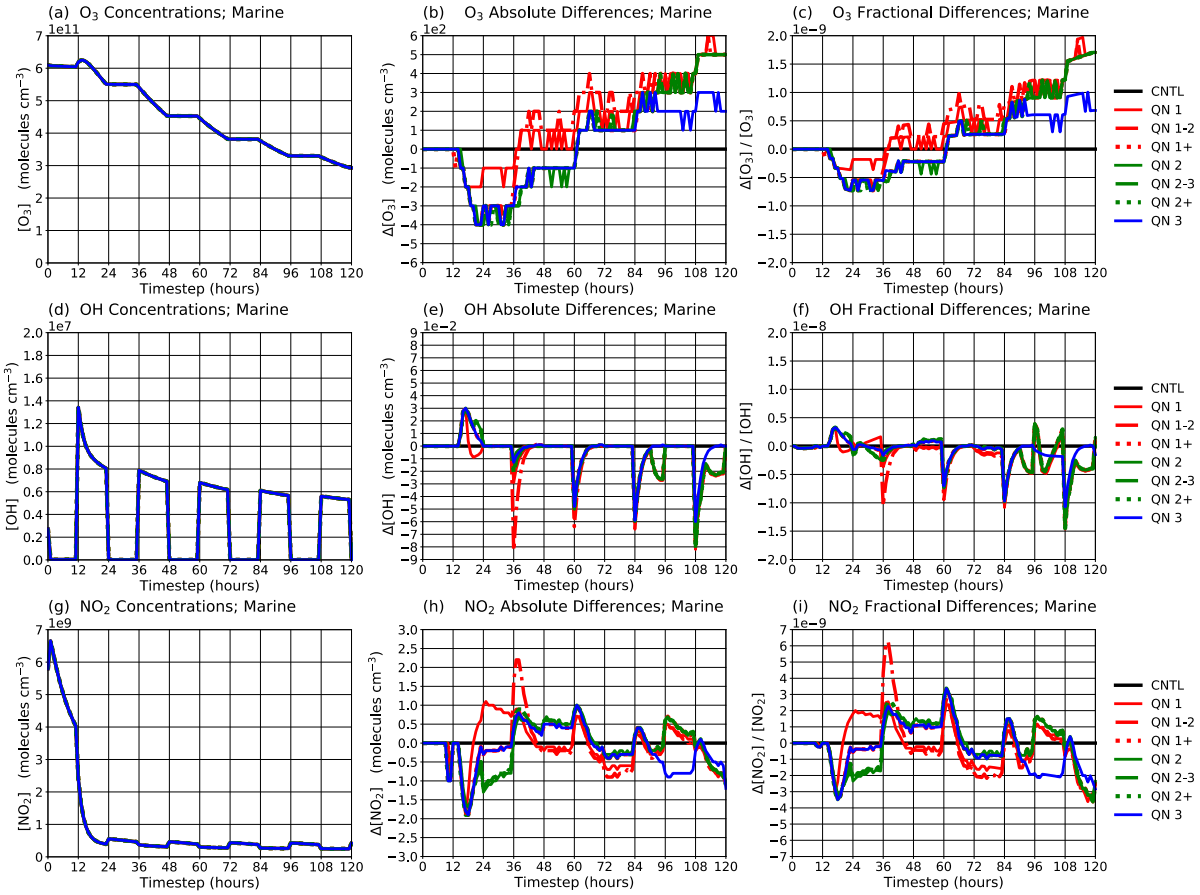


Figure S3. Same as Figure 2 in main paper, but for the Marine scenario

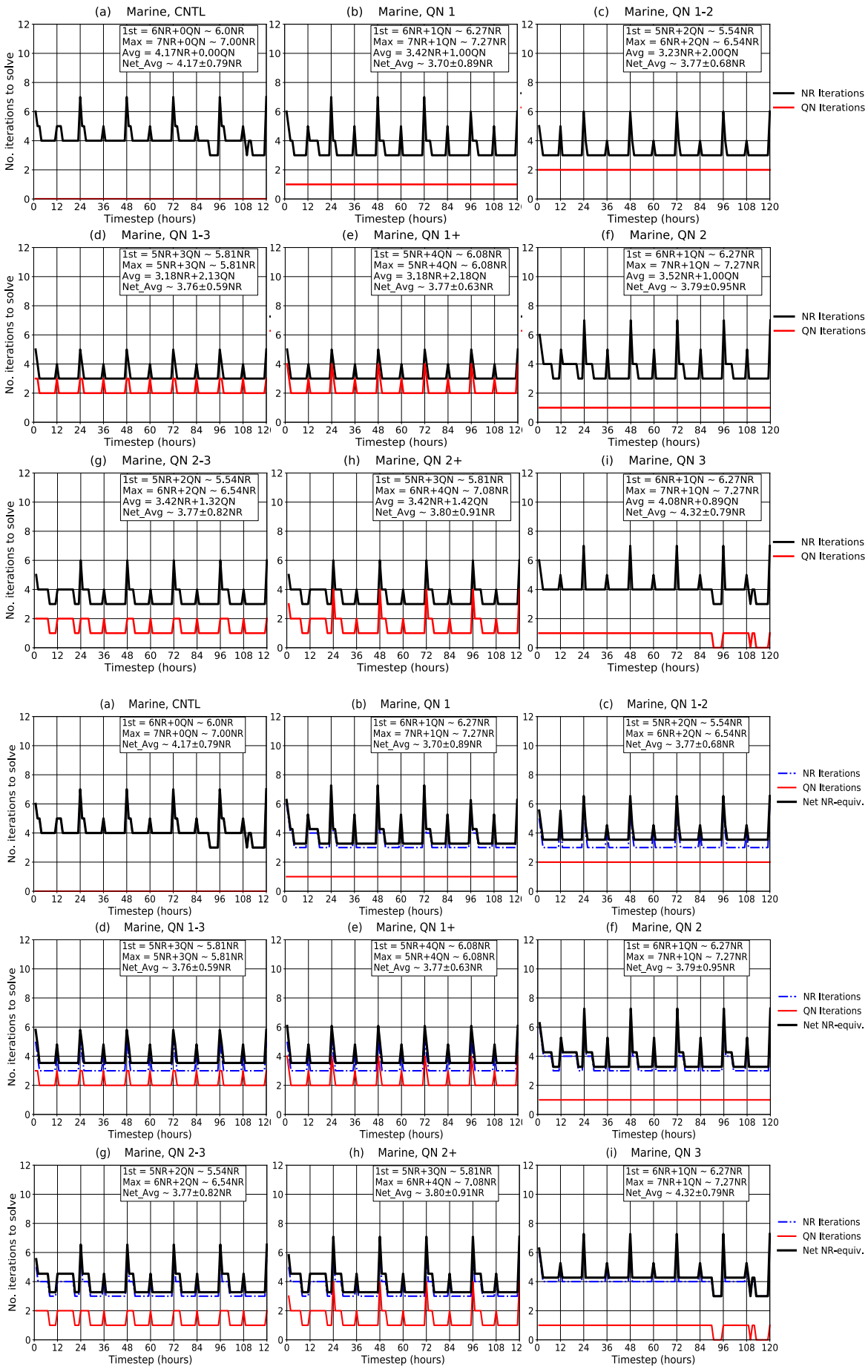

(a) Marine, CNTL
1st = 6NR+0QN ~ 6.0NR
Max = 7NR+0QN ~ 7.00NR
Avg = 4.17NR+0.00QN
Net_Avg ~ 4.17±0.79NR
(b) Marine, QN 1
1st = 6NR+1QN ~ 6.27NR
Max = 7NR+1QN ~ 7.27NR
Avg = 3.42NR+1.00QN
Net_Avg ~ 3.70±0.89NR
(c) Marine, QN 1-2
1st = 5NR+2QN ~ 5.54NR
Max = 6NR+2QN ~ 6.54NR
Avg = 3.23NR+2.00QN
Net_Avg ~ 3.77±0.68NR
(d) Marine, QN 1-3
1st = 5NR+3QN ~ 5.81NR
Max = 5NR+3QN ~ 5.81NR
Avg = 3.18NR+2.13QN
Net_Avg ~ 3.76±0.59NR
(e) Marine, QN 1+
1st = 5NR+4QN ~ 6.08NR
Max = 5NR+4QN ~ 6.08NR
Avg = 3.18NR+2.18QN
Net_Avg ~ 3.77±0.63NR
(f) Marine, QN 2
1st = 6NR+1QN ~ 6.27NR
Max = 7NR+1QN ~ 7.27NR
Avg = 3.52NR+1.00QN
Net_Avg ~ 3.79±0.95NR
(g) Marine, QN 2-3
1st = 5NR+2QN ~ 5.54NR
Max = 6NR+2QN ~ 6.54NR
Avg = 3.42NR+1.32QN
Net_Avg ~ 3.77±0.82NR
(h) Marine, QN 2+
1st = 5NR+3QN ~ 5.81NR
Max = 6NR+4QN ~ 7.08NR
Avg = 3.42NR+1.42QN
Net_Avg ~ 3.80±0.91NR
(i) Marine, QN 3
1st = 6NR+1QN ~ 6.27NR
Max = 7NR+1QN ~ 7.27NR
Avg = 4.08NR+0.89QN
Net_Avg ~ 4.32±0.79NR
No. iterations to solve
Timestep (hours)
NR Iterations
QN Iterations
Net NR-equiv.

Figure S4. Same as Figure 3 in main paper, but for the Marine scenario

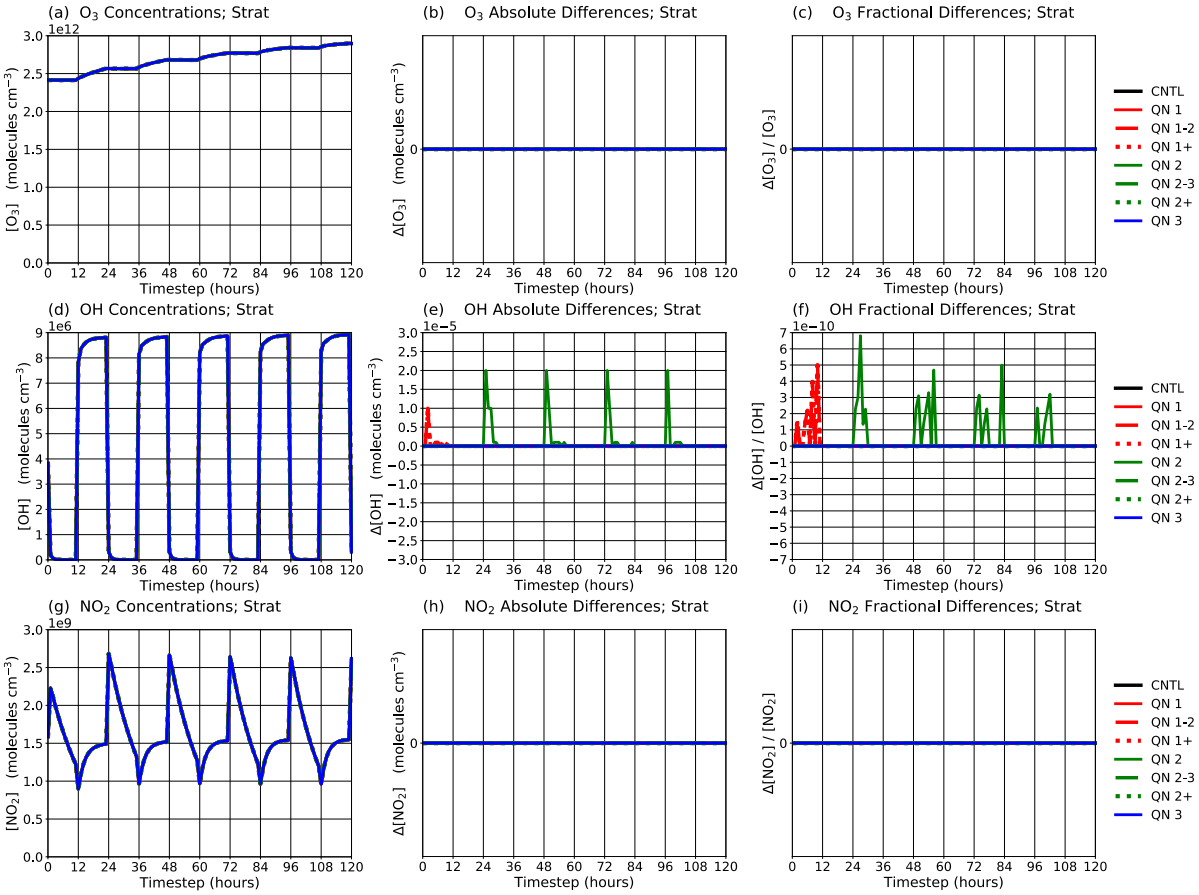


Figure S5. Same as Figure 2 in main paper, but for the Strat scenario

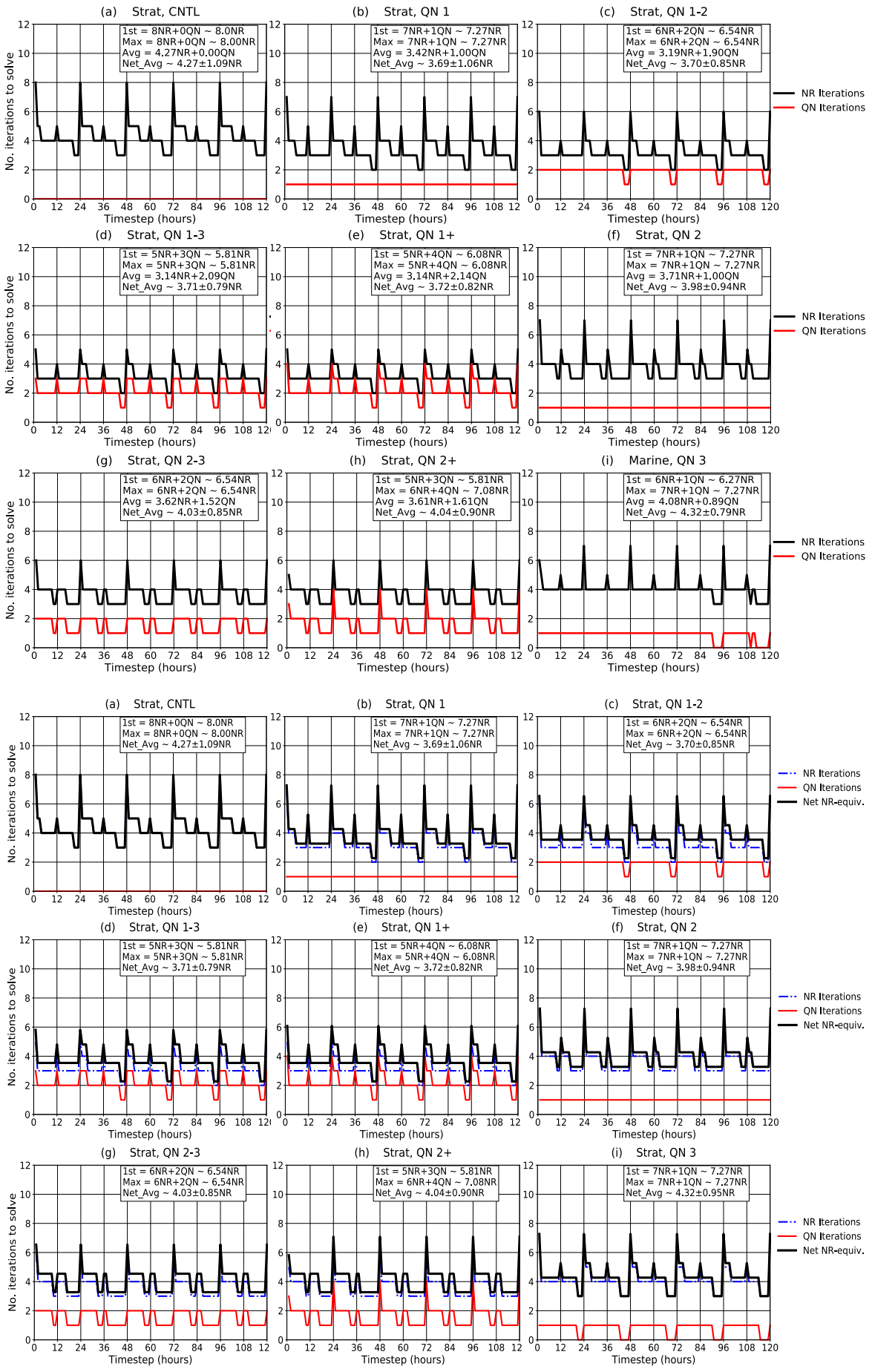

(a) Strat, CNTL
1st = 8NR+0QN ~ 8.0NR
Max = 8NR+0QN ~ 8.00NR
Avg = 4.27NR+0.00QN
Net_Avg ~ 4.27±1.09NR
(b) Strat, QN 1
1st = 7NR+1QN ~ 7.27NR
Max = 7NR+1QN ~ 7.27NR
Avg = 3.42NR+1.00QN
Net_Avg ~ 3.69±1.06NR
(c) Strat, QN 1-2
1st = 6NR+2QN ~ 6.54NR
Max = 6NR+2QN ~ 6.54NR
Avg = 3.19NR+1.90QN
Net_Avg ~ 3.70±0.85NR
(d) Strat, QN 1-3
1st = 5NR+3QN ~ 5.81NR
Max = 5NR+3QN ~ 5.81NR
Avg = 3.14NR+2.09QN
Net_Avg ~ 3.71±0.79NR
(e) Strat, QN 1+
1st = 5NR+4QN ~ 6.08NR
Max = 5NR+4QN ~ 6.08NR
Avg = 3.14NR+2.14QN
Net_Avg ~ 3.72±0.82NR
(f) Strat, QN 2
1st = 7NR+1QN ~ 7.27NR
Max = 7NR+1QN ~ 7.27NR
Avg = 3.71NR+1.00QN
Net_Avg ~ 3.98±0.94NR
(g) Strat, QN 2-3
1st = 6NR+2QN ~ 6.54NR
Max = 6NR+2QN ~ 6.54NR
Avg = 3.62NR+1.52QN
Net_Avg ~ 4.03±0.85NR
(h) Strat, QN 2+
1st = 5NR+3QN ~ 5.81NR
Max = 6NR+4QN ~ 7.08NR
Avg = 3.61NR+1.61QN
Net_Avg ~ 4.04±0.90NR
(i) Marine, QN 3
1st = 6NR+1QN ~ 6.27NR
Max = 7NR+1QN ~ 7.27NR
Avg = 4.08NR+0.89QN
Net_Avg ~ 4.32±0.79NR
No. iterations to solve
Timestep (hours)
NR Iterations
QN Iterations
(a) Strat, CNTL
1st = 8NR+0QN ~ 8.0NR
Max = 8NR+0QN ~ 8.00NR
Net_Avg ~ 4.27±1.09NR
(b) Strat, QN 1
1st = 7NR+1QN ~ 7.27NR
Max = 7NR+1QN ~ 7.27NR
Net_Avg ~ 3.69±1.06NR
(c) Strat, QN 1-2
1st = 6NR+2QN ~ 6.54NR
Max = 6NR+2QN ~ 6.54NR
Net_Avg ~ 3.70±0.85NR
(d) Strat, QN 1-3
1st = 5NR+3QN ~ 5.81NR
Max = 5NR+3QN ~ 5.81NR
Net_Avg ~ 3.71±0.79NR
(e) Strat, QN 1+
1st = 5NR+4QN ~ 6.08NR
Max = 5NR+4QN ~ 6.08NR
Net_Avg ~ 3.72±0.82NR
(f) Strat, QN 2
1st = 7NR+1QN ~ 7.27NR
Max = 7NR+1QN ~ 7.27NR
Net_Avg ~ 3.98±0.94NR
(g) Strat, QN 2-3
1st = 6NR+2QN ~ 6.54NR
Max = 6NR+2QN ~ 6.54NR
Net_Avg ~ 4.03±0.85NR
(h) Strat, QN 2+
1st = 5NR+3QN ~ 5.81NR
Max = 6NR+4QN ~ 7.08NR
Net_Avg ~ 4.04±0.90NR
(i) Strat, QN 3
1st = 7NR+1QN ~ 7.27NR
Max = 7NR+1QN ~ 7.27NR
Net_Avg ~ 4.32±0.95NR
No. iterations to solve
Timestep (hours)
NR Iterations
QN Iterations
Net NR-equiv.

Figure S6. Same as Figure 3 in main paper, but for the Strat scenario

2. ADDITIONAL UM-UKCA PLOTS and ANALYSIS

a) QN Performance over 20-year run

The following plot gives a neat visual of the performance of the QN method for the 20-year run on 432 cores.

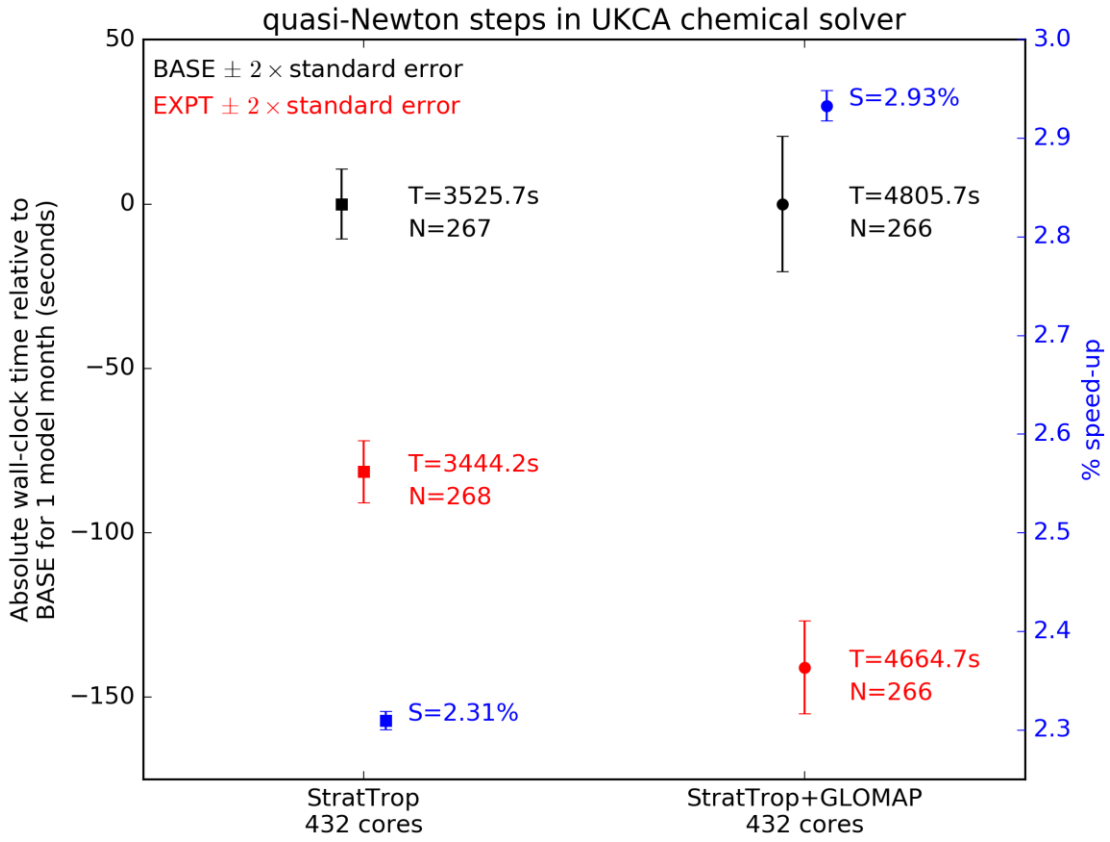


Figure S7. UM speed up for the StratTrop and StratTrop-GLOMAP modes with CNTL and QN implemented runs

b) Time Dependence of Differences Between Simulations and Bias

For the results to be reliable not only we need to show high accuracy (small difference) as demonstrated in the article but also that the differences between the modified and original simulations do not grow and stay bounded in time.

The first set of three plots (S8-S10) below show, for the OH, normalised absolute mean difference (NMAD), normalised root mean square difference (NRMSD) and normalised mean bias (NMB). The second set of three plots show the same quantities for the Ozone.

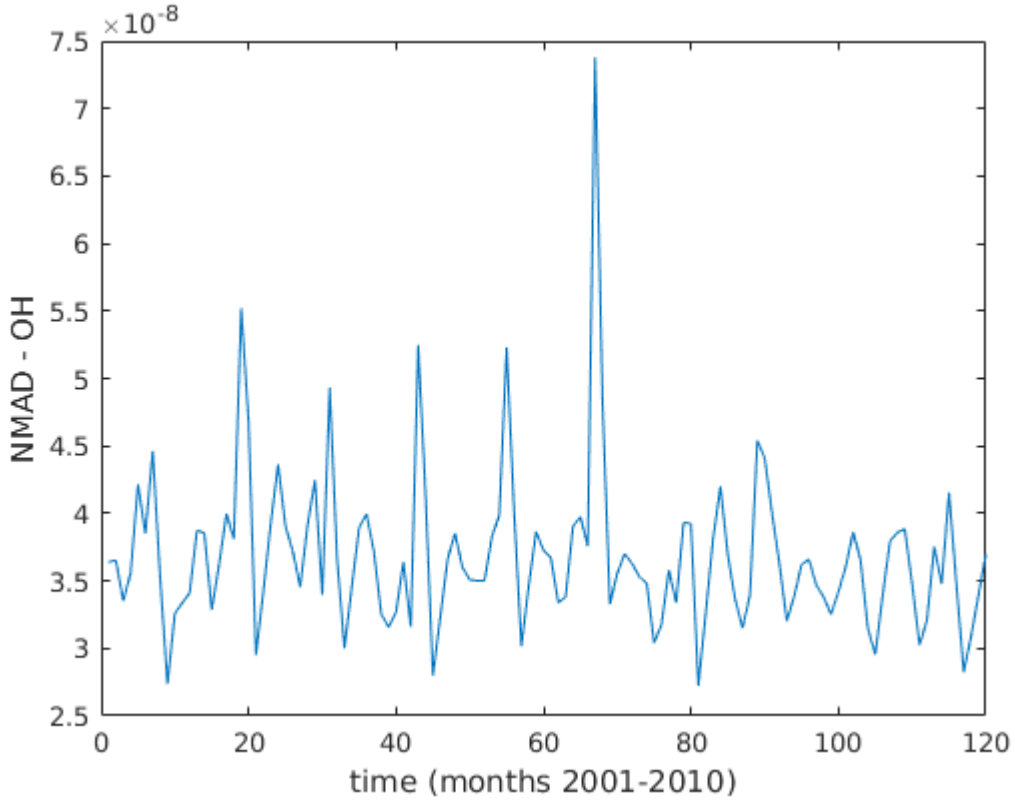


Figure S8. NMAD *for monthly averaged OH concentration for the StratTrop Run (Years 2001-2010).*

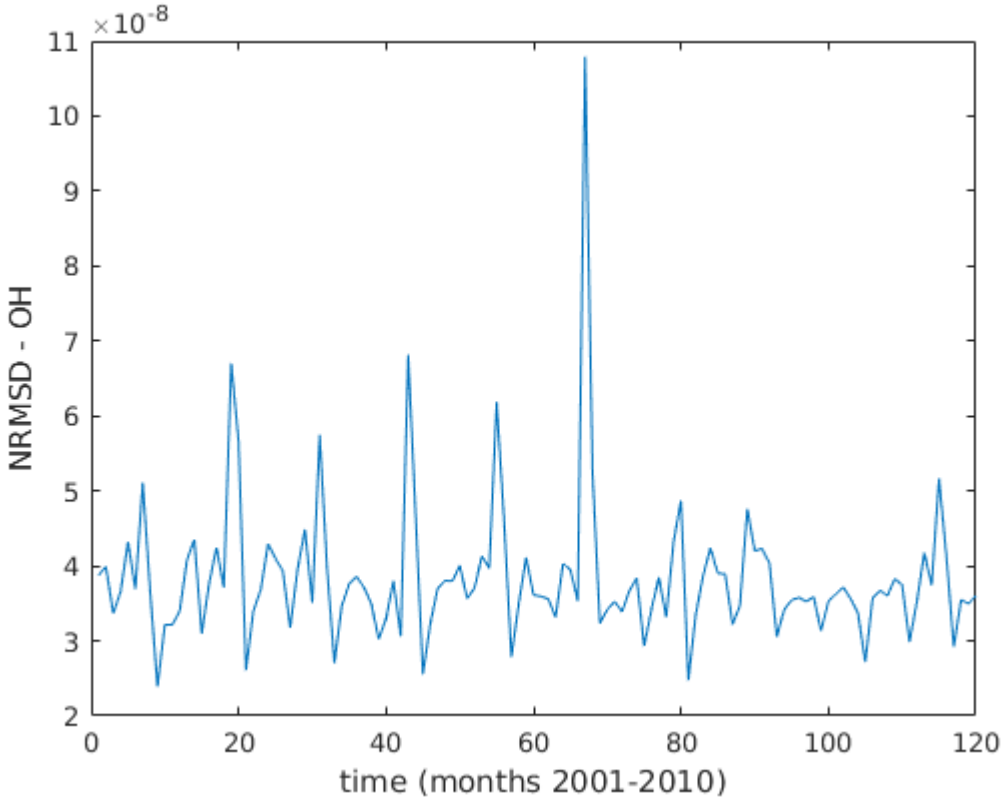


Figure S9. NRMSD for monthly averaged OH concentration for the StratTrop Run (Years 2001-2010).

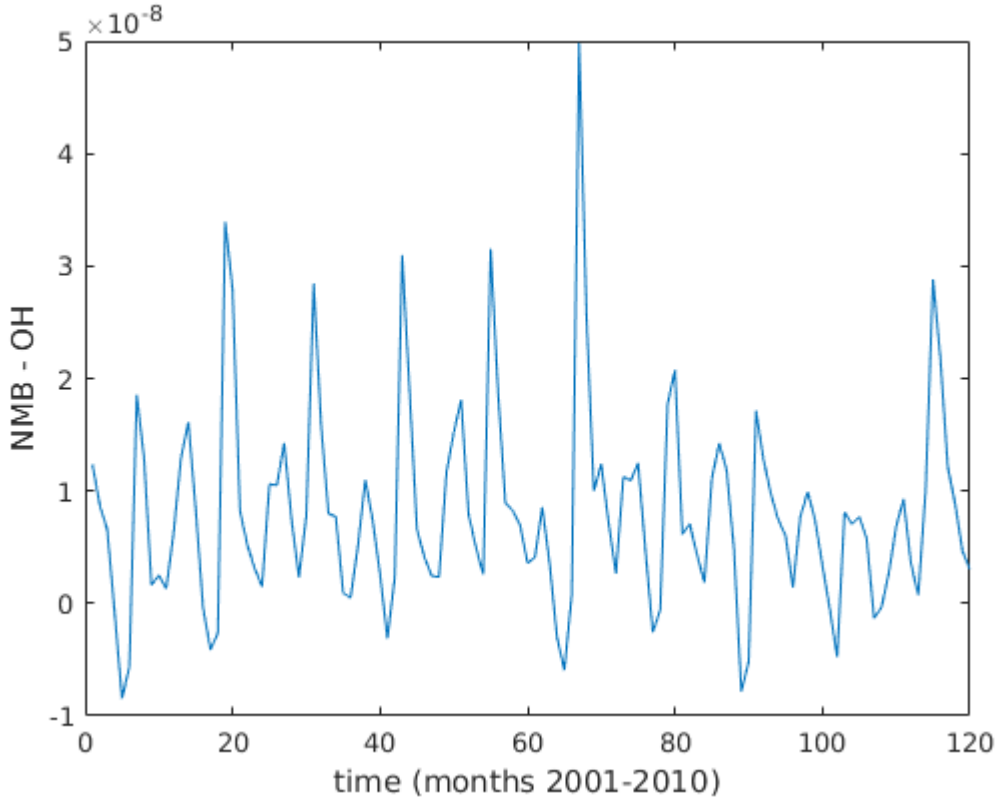


Figure S10. NMB for monthly averaged OH concentration for the StratTrop Run (Years 2001-2010).

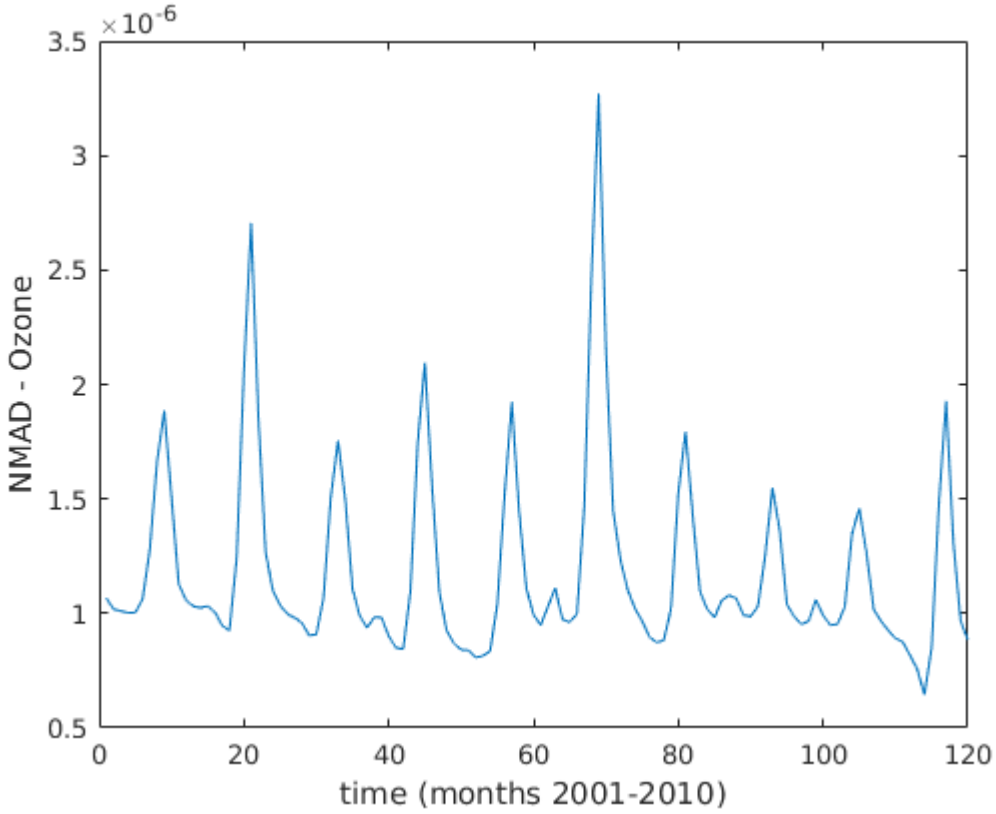


Figure S11. NMAD of monthly averaged Ozone concentration for the StratTrop Run (Years 2001-2010).

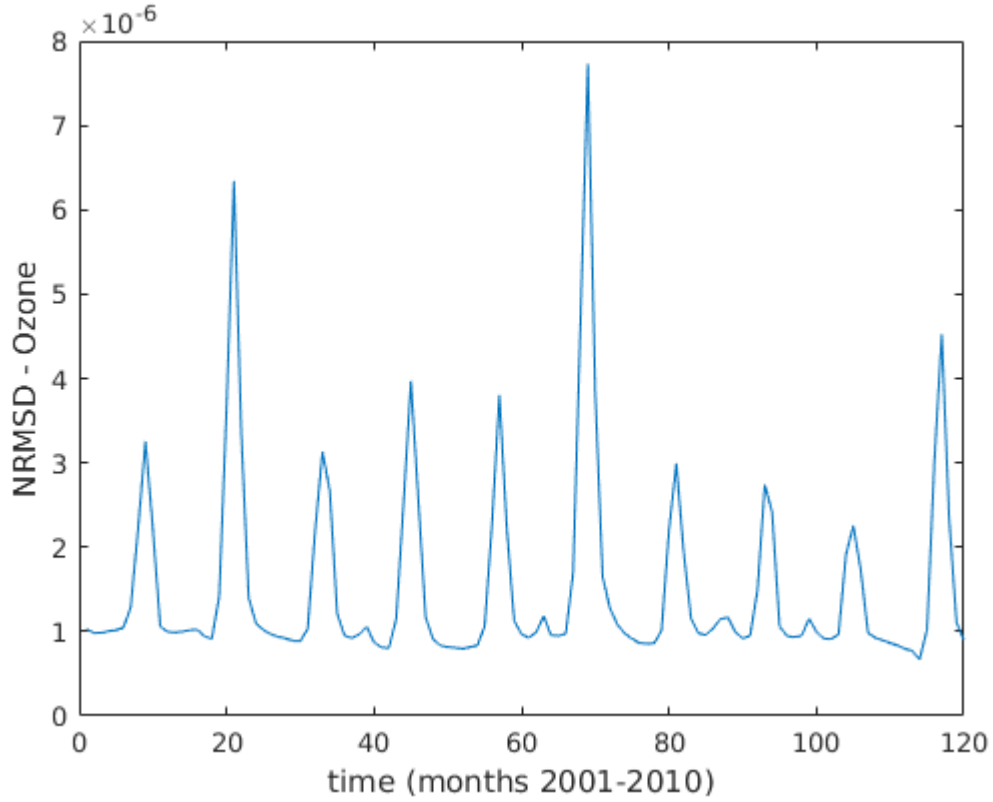


Figure S12. NRMSD of monthly averaged Ozone concentration for the StratTrop Run (Years 2001-2010).

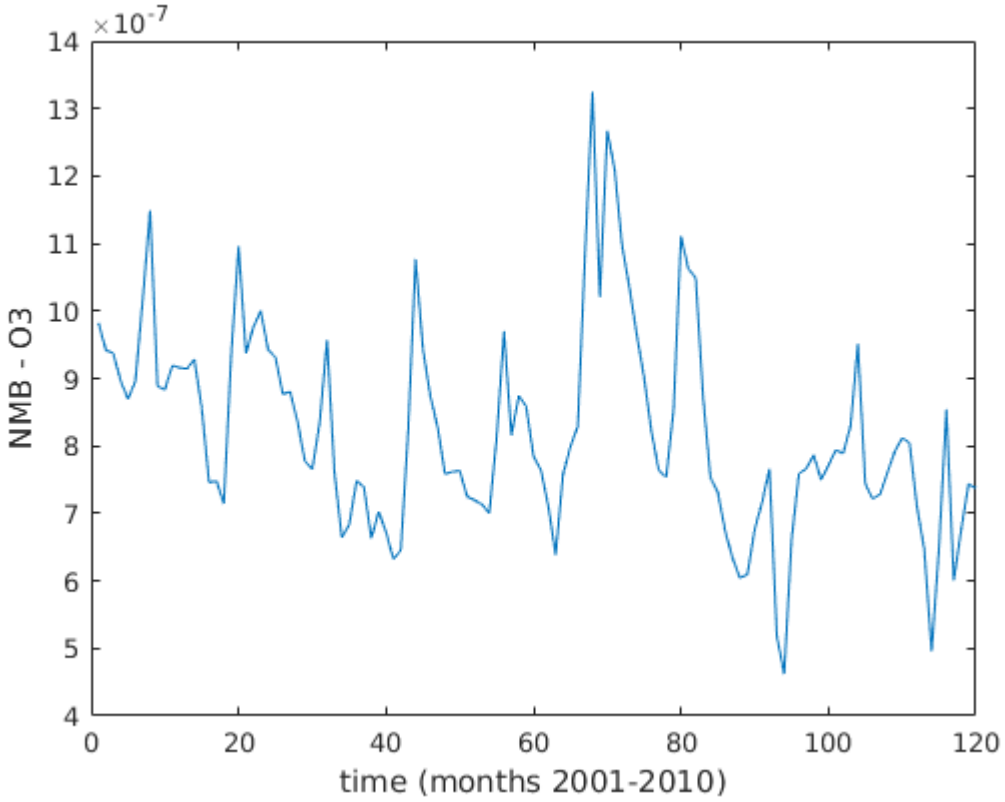


Figure S8. NMB of monthly averaged ~~OH~~ Ozone concentration for the StratTrop Run (Years 2001-2010).

c) NMAD, NRMSD and NMB Values for all Species

Below, we give a full table of NMAD, NRMSD and NMB values of all species for the UM-UKCA global model.

Species	NMAD (*E-3) After 120 months	NMAD (*E-3) After 240 months	NRMSD (*E-3) After 120 months	NRMSD (*E-3) After 240 months	NMB (*E-3) After 120 months	NMB (*E-3) After 240 months
O3	0.0011	0.00088	0.0010	0.00090	0.00098	0.00074
NO	0.0441	0.0340	0.0066	0.0077	-0.0032	-0.0020
NO3	0.0027	0.0040	0.0074	0.0170	-0.0011	-0.0021
N2O5	0.0045	0.0027	0.0052	0.0074	-0.0037	-0.0016
HO2NO2	0.0053	0.0059	0.0079	0.0086	-0.0013	-0.0015
HNO3	0.0066	0.0058	0.0074	0.0116	-0.0046	-0.0025
H2O2	0.0150	0.0172	0.0332	0.0423	-0.0135	-0.0158
CH4	0.0003	0.0003	0.0004	0.0004	0.0003	0.0003
CO	0.0024	0.0025	0.0019	0.0020	0.0017	0.0017
HCHO	0.0089	0.0101	0.0276	0.0343	-0.0033	-0.0050
CH3OOH	0.0089	0.0105	0.0180	0.0220	-0.0049	-0.0064
HONO	0.0206	0.0204	0.1723	0.2035	0.0017	0.0040
C2H6	0.0118	0.0137	0.0098	0.0129	0.0118	0.0136
C2H5OOH	0.0156	0.1939	0.0186	0.0279	0.0085	0.0111
CH3CHO	0.0098	0.0012	0.0010	0.0013	0.0056	0.0067
PAN	0.0115	0.0153	0.0319	0.0530	-0.0011	-0.0006
C3H8	0.0092	0.0127	0.0071	0.0140	0.0089	0.0123
n-C3H7OOH	0.0140	0.0225	0.0189	0.0412	0.0103	0.0180
i-C3H7OOH	0.0152	0.0237	0.0198	0.0412	0.0103	0.0180
C2H5CHO	0.0093	0.0134	0.0408	0.0373	0.0072	0.0101
C2H6CO	0.00933	0.01099	0.00783	0.00959	0.00926	0.01086
CH3COCH2OOH	0.01394	0.01660	0.01950	0.02661	0.00301	0.00494
PPAN	0.01001	0.01193	0.03702	0.06292	-0.00931	-0.01031

CH3ONO2	0.01053	0.01355	0.01481	0.01974	-0.00932	-0.01286
C5H8	0.02416	0.02421	0.02180	0.02464	0.01083	0.01182
ISOOH	0.02798	0.02933	0.03170	0.03713	0.01167	0.01110
ISON	0.03209	0.04159	0.06750	0.10933	0.00375	0.01014
MACR	0.02345	0.02557	0.02783	0.03507	0.01884	0.01998
MACROOH	0.04468	0.05189	0.05149	0.07872	0.02563	0.03051
MPAN	0.03073	0.03666	0.04900	0.06991	-0.00697	-0.00842
HACET	0.02630	0.03193	0.03694	0.05458	-0.00538	-0.00461
MGLY	0.14927	0.18045	0.41070	0.63219	-0.14196	-0.17186
NALD	0.04737	0.06276	0.10608	0.17306	0.02292	0.04034
HCOOH	0.01836	0.02537	0.02768	0.04264	-0.00854	-0.00811
CH3CO3H	0.02581	0.03148	0.03700	0.05843	0.01038	0.01493
CH3CO2H	0.02374	0.02903	0.03652	0.05757	-0.00118	0.00231
MVKOOH	0.02997	0.03284	0.05591	0.06614	0.00713	0.00526
Cl	0.00069	0.00035	0.00090	0.00036	-0.00069	-0.00020
ClO	0.00245	0.00247	0.00738	0.00731	0.00015	0.00064
Cl2O2	0.09651	0.05123	0.28837	0.05308	0.00543	0.02153
OClO	0.01277	0.01082	0.01654	0.01336	0.00894	0.00849
Br	0.00051	0.00056	0.00048	0.00065	-0.00029	-0.00023
BrCl	0.01104	0.00941	0.01503	0.01262	0.00771	0.00731
BrONO2	0.00471	0.00439	0.00757	0.00754	-0.00234	-0.00186
N2O	0.00004	0.00003	0.00008	0.00005	0.00004	0.00003
HOCl	0.00639	0.00587	0.01665	0.01657	0.00087	0.00045
HBr	0.008348	0.00908	0.01627	0.015961	-0.00754	-0.0083
HOBr	0.006061	0.00595	0.011883	0.012055	0.00258	0.00168
ClONO2	0.001965	0.00217	0.00272	0.0031523	-8E-05	0.00043
CFCl3	4.35E-05	3.9E-05	8.4E-05	7.53E-05	4.3E-05	3.8E-05
CF2Cl2	3.54E-05	2.7E-05	7E-05	5.03E-05	3.3E-05	2.5E-05
CH3Br	0.00044	0.00047	0.00052	0.00055	0.00044	0.00047
N	0.000798	0.00403	0.0006	0.004242	-0.00077	-0.0027
O(3P)	2.66E-05	1.3E-05	4.6E-05	3.19E-05	-1.9E-05	-5E-06
ORGNIT	0.056804	0.06181	0.08377	0.107436	0.03981	0.04092

H	8.77E-06	5.4E-06	2.7E-05	1.99E-05	8.8E-06	2.2E-06
OH	3.64E-05	3.7E-05	3.9E-05	3.6E-05	1.2E-05	3E-06
HO2	0.001041	0.00108	0.00109	0.001262	0.0003	0.00023
CH3OO	0.00291	0.00492	0.00214	0.003857	0.0025	0.00375
C2H5OO	0.056128	0.0569	0.16649	0.16476	0.04521	0.04526
CH3CO3	0.051005	0.05548	0.21106	0.274743	-0.02183	-0.0224
n-C3H7OO	0.076912	0.08898	0.2034	0.272403	0.07119	0.08267
i-C3H7OO	0.070862	0.08225	0.19265	0.258701	0.06177	0.07211
C2H5CO3	0.032232	0.04066	0.11543	0.191884	-0.01635	-0.0173
CH3COCH2OO	0.043401	0.04446	0.11144	0.119976	0.01727	0.0185
CH3OH	0.005674	0.00593	0.007	0.009182	-0.00167	-0.003
HCl	0.000445	0.00049	0.00135	0.001333	-8.3E-05	-0.0001
BrO	0.001422	0.00155	0.00471	0.004844	-0.00062	-0.0006
NO2	0.004671	0.00346	0.00718	0.012697	-0.00346	-0.002
O(1D)	2.69E-05	1.8E-05	3.2E-05	2.43E-05	1.2E-05	1.1E-05