24 More Years of Numerical Weather Prediction
A model performance model

Gerard Cats

Author: Cats, G. J.

● ● ● ●

24 More Years of Numerical Weather Prediction: A Model Performance Model

Gerard Cats

May 19, 2008

Abstract

For two formulations of currently usual numerical weather prediction models the evolution of maximum achievable resolution is predicted for the next 24 years. One of those formulations is the semi-implicit, semi-Lagrangian and spectral (sisLsp) method as used *e.g.,* by ECMWF. This formulation requires global communications. The other one is the explicit, Eulerian, grid point (exEugp) formulation, requiring local communications only. The evolution prediction is based on an analysis of computation and communication requirements and on extrapolation of computation and communication performance of future computer systems. The main result is that model performance will continue to grow, approximately exponentially, for another two decades. After that, however, the finite speed of light will make global communications very inefficient. As a result, towards the end of our investigation period, sisLsp models lose their current advantage, which derives from their lower compute operations count, and exEugp models start to perform better, because they do not need global communications. The extrapolation of computer performance into the future requires quite a few assumptions on future configurations. Our results are not very sensitive to those assumptions, except if the configuration parameters are chosen to model a Grid computer. On a Grid computer, even models without global communication requirements show some performance loss due to the finiteness of the speed of light towards the end of the investigation period.

1   Introduction

Around 1984 parallel computing started to make its way into Numerical Weather Prediction (NWP) [14]. First experiments were with a explicit Eulerian grid point model (denoted by exEugp). By the time that parallel computers were developing into massively parallel, distributed memory systems, the NWP models had developed into semi-implicit semi-Lagrangian spectral models (denoted by sisLsp). An important reason to move from exEugp to sisLsp was the almost two orders of magnitude reduction in computation operations count for the same problem size. Implementation of an explicit Eulerian grid point model on a massively parallel system is relatively simple, but implementation of semi-implicit, spectral or semi-Lagrangian methods is rather more complex. The increasing complexity of the use of distributed memory systems is nicely demonstrated by a series of papers on HIRLAM [5], a limited area NWP model: [40] on the explicit Eulerian grid point version, [41] on the spectral, and [42] on the semi-Lagrangian version of HIRLAM. The main reason for the increased complexity is that an explicit Eulerian grid point formulation needs nearest-neighbour communication only, whereas a semi-implicit model requires the solution of a Helmholtz equation, with global communication; a spectral model needs Fourier transforms, also with global communications, and a semi-Lagrangian model needs local communications, but not necessarily restricted to nearest neighbours (*e.g.,* perhaps also with neighbours of neighbours).

With increasing computation speed, and with increasing number of processors, it has often been surmised that sooner or later global communications would become so expensive, that communication overhead would completely dominate the calculations. Because local communications could be much cheaper, at that stage in time it might be cheaper to run a exEugp model, even though the operations count in such a model is much

higher than in a sisLsp model. The aim of this paper is to investigate whether indeed sooner or later we are going to see this happening.

For this aim, we basically are going to extrapolate recently observed trends in computer computation and communication speeds, in number of processors, and physical size of the computer. For each of the years until 2032 we thus find a computer configuration that might be available for NWP. We will develop a simple model, relating the compute and communication burden of a NWP model to the number of grid points of that model. We then assume that model calculations must complete within a certain, fixed, time slot (we took 1 hour). This then sets the maximum number of grid points that can be processed on that computer. We are going to use that number of grid points as a measure for the quality of the model. In this way, we can determine for each year what model will be the "best" that can be handled.

Throughout the paper, we will heavily lean on the situation at ECMWF. This is mainly because the ECMWF system is at the fore-front of NWP and well-documented. The ECMWF system is global, and spectral (sisLsp). Yet we will take several concepts from grid point limited area models, like the number of grid points in the x-direction. Even though our considerations lean on the ECMWF situation, the results of our calculations do not. In the end, the ECMWF model is mainly used to normalise the relative contributions of the different steps, like computations, local and global communications. Hence, we never worry about proportionality factors. Only in the final stages will they be needed, and then they will be derived from ECMWF data.

In general we are going to see that the sisLsp model will beat the exEugp model for almost two decades to come. ECMWF bought its first parallel computer, a Cray X-MP, in 1984. Now, 24 years later , we are going to look ahead for 24 years. In the last few years of that time frame, we are going to see a total breakdown of sisLsp computing, but exEugp will show a continued performance growth.

Twenty-four years is a long time to look ahead. It is likely that our results will strongly depend on assumptions we made on compute and communication performance. Yet, by varying the assumptions we found that the conclusions do not change substantially, unless the parameters are chosen to represent a Grid computer.

In the next section of this paper we will describe the development of NWP models from the straightforward exEugp to the currently most sophisticated sisLsp. The focus of that section is to describe the relative efficiency of the different model formulations, both for computations and for communications. From that it will be clear that it is useless to use the semi-Lagrangian formulation in an explicit model. So even though an explicit semi-Lagrangian method takes an intermediate position in communication requirements between a Eulerian explicit and a Eulerian semi-implicit formulation, we will not investigate such constructs at all. Only the two extremes will be studied: exEugp and sisLsp.

Many of the subsequent sections of this paper will be dominated by assumptions we have to make on the developments of the NWP system and of computers, and on how a NWP system is mapped onto a computer. To give the reader a ahead warning for the many assumptions, we inserted a special section, describing the main assumptions.

In Section 4, we develop a simple model for execution time. We estimate the time spent in computations, (apart from a proportionality factor), and the time spent in communications, based on the number and size of the messages. This model is used in Section 5 to study how the different terms contributing to the NWP model execution time scale with the number of processors. For the rest of this paper, this particular section mainly serves to show that memory access rate does not depend on the number of processors. Because in current NWP models hardly any time is lost in memory access, and assuming that memory access will not become slower in the future, we do not have to separate memory access time out from the computation time.

In Section 6 we derive empirically, from results obtained in the past, how compute and communication performance, and how the number of processors will improve in time. Also, from the ECMWF model we derive the proportionality factors of the different terms contributing to total execution time. We normalise these factors to have a total execution time of 1 hour

Section 7 presents the results; first from the 'baseline' configuration, then from our model after varying several

parameters. One of the variations we examine allows us to draw conclusions on the performance of NWP models on Grid computers.

In all sections up to and including Section 7 we essentially assumed NWP and computer model developments by extrapolating observed trends in the recent past. This is the 'business as usual' model. In the Discussion we will give some attention to possible other developments. In an Appendix we write out the main sets of equations that determine the computational and communication requirements of NWP models: the primitive equations and the Helmholtz equation.

2   A historical hierarchy of NWP models

Probably V. Bjerknes [2] was the first to realise that atmospheric flow obeys the laws of classical fluid dynamics. This resulted in a set of equations for the atmospheric variables like pressure, wind direction and speed, temperature. Richardson's [31] application of these equations is world-famous, in spite of its failure to produce reasonable results.

Charney [6] made a number of approximations to the equations, with which he was able to make a one day weather forecast. But Charney added the statement that in the course of time these approximations would probably be abandoned.

The main (and good) approximation was to assume that the atmospheric flow is balanced, *i.e.,* that the wind is proportional and perpendicular to the pressure gradient. This reduced the number of independent variables, and thus computation time. A second approximation was to assume that the atmosphere was two-dimensional, hence vertical variations are neglected. Again, this reduces the number of independent variables. Such a model is called barotropic. A third approximation was to assume the flow to be in hydrostatic balance. In this approximation, vertical velocities are allowed, but vertical accelerations are neglected. This again results in fewer independent variables, but the main computational gain is achieved by the fact that the resulting equations are less complex. The Boussinesq approximation [19] is common to all models. It removes sound waves from the solution space of the models.

In numerical weather prediction (NWP) systems the barotropic approximation was dropped in favour of multi-level (so called baroclinic) models in the fifties of the 20th century. The balanced equations became replaced by the primitive equations in the seventies. The hydrostatic approximation is still very common; only models that run at very high resolution (MM5 [11], Aladin [1], LokalModell [39]) nowadays are based on the full, non-hydrostatic, equations.

Within the currently common NWP models (be it hydrostatic or not) there is a variety of integration methods, like explicit/semi-implicit, Eulerian/semi-Lagrangian, and grid point/spectral methods. For each we will mention the main properties determining the computation and communication requirements.

For total execution time, an essential consideration will be the CFL criterion. This criterion puts an upper bound to the size of the time step of the model:

$$\Delta t \leq \kappa \Delta x / c \,, \tag{1}$$

where Δx is the horizontal grid distance and c the speed of the fastest waves sustained by the equations. The constant κ depends on the time integration scheme, but is always of order unity [13]. Integrations with larger time steps than Equation (1) are unstable. The dimensionless quantity $c\frac{\Delta t}{\Delta x}$ is called the Courant number. To make a weather forecast of a certain length, say 1 day, the number of time steps taken by the model, and hence the total execution time, is inversely proportional to the size of the time step. The variety of integration methods is mainly the consequence of the attempts to allow larger time steps, *i.e.,* to reduce the speed of the fastest waves in the model. We will base the following of this section on the linearised primitive equations [22].

Our analysis of communication requirements will assume that the model is implemented on a massively parallel machine, with straightforward domain decomposition for the grid point communications (see Figure (1)).

explicit versus semi-implicit: Due to the presence of the so-called gradient terms the linearised primitive equa-
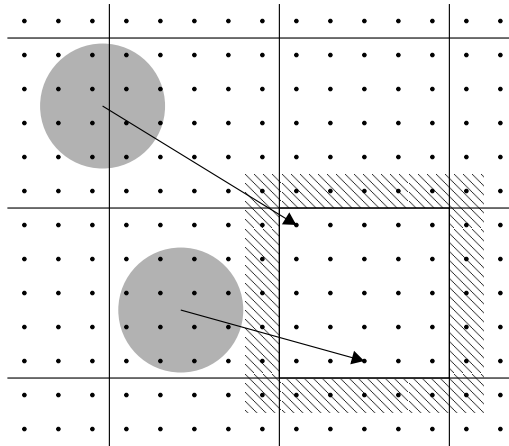
Figure 1: Domain decomposition, where each processor processes 25 grid points. Communication patterns: Eulerian advection requires communication from a halo zone, in the Figure the hatched band of one grid point width. Semi-Lagrangian advection for two grid points is depicted by arrows; the shaded circle around the tail of the arrow contains the grid points that for a certain interpolation scheme may influence the grid point at the arrow head.

tions sustain gravity waves, travelling at a speed of approximately 300 m/s (Appendix (A.1)). The equations can be solved with an explicit scheme, but the CFL criterion becomes very restrictive. For example, if the grid distance is 60 km, sufficient for the description of many extra-tropical weather systems, the maximum time step would be in the order of 200 s. This is far smaller than the time scale at which those systems change (1 hour). So the CFL criterion puts a far heavier constraint on the time step than the accuracy of the time discretisation would require.

It is possible to soothe the severeness of the CFL criterion by more complicated time integration schemes, allowing larger κ in Equation (1). Durran [13] lists a number of those schemes; *e.g.,* the fourth order Runge-Kutta scheme allows Courant numbers up to 2.8. However, this comes at the expense of a more damping and more costly integration scheme. In our investigations later on, we will assume that an explicit model can be run with a κ of 1.5, as a balance between larger time step on the pro-side, and more computations per time step as a con.

Even much more costly is the implicit method, but it has the advantage of being unconditionally stable: the time step is not limited by stability, but only by the required physical accuracy.

By treating the gradient terms implicitly, the time step is no longer determined by the gravity wave speed. Because other terms in the equations are still treated explicitly, this scheme is called semi-implicit. The explicit terms include the advection terms. The advection speed is the (maximum) wind speed, which is in the order of 100 m/s. Hence, the time step size with a semi-implicit scheme is up to three times that of an explicit scheme. An explicit scheme has a very simple communication pattern: only nearest-neighbour communications are needed. A semi-implicit scheme, however, requires to solve a number of two-dimensional Helmholtz equations (Appendix (A.2)), for which global communications are necessary. The additional pure computation time for the semi-implicit scheme, in particular to solve the Helmholtz equations, is negligible. So on a single processor machine, where no communication is needed, the semi-implicit scheme, with its 3 times larger time step, saves a factor of 3 over an explicit scheme. Finally, it is remarked that in non-hydrostatic models a similar semi-implicit treatment results in a three-dimensional Helmholtz equation. Its solution may be so costly that explicit methods may be competitive even on single processor machines [38].

Eulerian versus semi-Lagrangian: After sound waves and gravity waves, the next fastest waves are those due to the advection terms in the primitive equations. For these, the CFL criterion derives from the maximum wind speed (in the order of 100 m/s). It determines the time step in the standard solution method, called Eulerian.

However, advection can also be treated by following an air particle along its trajectory. This method, called Lagrangian, is unconditionally stable. In practice NWP models apply the Lagrangian method each time step, where for each grid point a trajectory is determined, ending at that grid point, and starting one time step earlier. The properties of the air particle, like temperature and wind, at the end point are taken equal to those at the start point of the trajectory. Because of this mixture of Lagrangian techniques with standard grid point modelling, the method is called semi-Lagrangian. The start point will usually not be on a grid point, so an interpolation between grid point values will be required. Also, the trajectory is only known after the wind has been determined, but the wind is dependent on the start point. So an iteration is required. As a consequence, the pure computation time for the semi-Lagrangian scheme per time step is notably more than in a Eulerian scheme, usually in the order of 20%. The communication pattern may be simple: if a trajectory start point, and in view of the required interpolation, the points surrounding it, are on a neighbouring processor, nearest neighbour communication will suffice. If, however, the trajectories span many grid points, *i.e.,* the Courant number is much larger than 1, and the domain covered by a processor contains few grid points, next-neighbour or even further away communication may be needed. In these circumstances, communication overhead may become substantial.

grid point versus spectral: The gradient terms can be calculated more accurately in spectral space. In other words, to achieve a similar accuracy, fewer spectral components are needed than grid points. A general rule of thumb is that a grid point model needs about twice as many grid points as a spectral method would need spectral components to achieve comparable accuracy [26]. This would result in lower computation costs. This advantage is reduced by the required spectral transformations. The costs of these nowadays is quite acceptable, mainly thanks to the use of efficient Fourier algorithms. Because the Helmholtz equation is easily solved in spectral space, spectral methods furthermore save on the computation time for the implicit scheme. The spectral transformations require global communications. This is a disadvantage with respect to the local communications needed by an explicit model, but once the choice is for semi-implicit, the spectral method does not introduce substantial further communication overhead.

In the course of time, the increased computer power has been used to increase horizontal and vertical resolution. Also, the models kept more and more variables. Initially the only atmospheric component was dry air, but nowadays models keep track of water, often in a variety of phases (vapour, liquid, ice, snow) and possibly chemical constituents like ozone.

An important inference from the above considerations is that it only makes sense to use semi-Lagrangian methods if the fastest waves, *i.e.,* the gravity waves, have already been eliminated, so in a semi-implicit scheme.

3   Assumptions

Inevitably, predictions on the development of (NWP) models and computers for NWP require assumptions on the future form of NWP models and the architecture of computers. In this section, we will cursorily treat the several assumptions. In the course of later sections, we will add more discussion as we need the assumptions; there we will include references to substantiating literature.

3.1   The NWP model

After initial use of approximated NWP models, the 1970-ies have seen the general acceptance of models based on the primitive equations. Since then, the models have become increasingly complex, mainly in terms of number of variables. Also the number of floating point operations per variable has increased; this was mainly to alleviate the constraints put by stability on the size of the time step. Because total execution time is almost proportional to the number of time steps, and hence inversely to the time step size, the past decades have seen much effort into increasing the time step size. In the most complex codes that currently are used in operational weather forecasting (sisLsp) the time step is not constrained anymore by stability, but merely by accuracy. Hence, it is unlikely that more complicated integration schemes will allow further increases in time

step, and therefore the number of floating point operations per model variable in those models will not further increase substantially. So we are going to assume that the complexity can be measured by the number of variables, without further allowance for more complex integration schemes. In contrast, in the most simple model formulation (exEugp) the model equations cannot become more complex - by definition. Hence also for these models the complexity can be measured by the number of variables.

In general, we assume that the model computation time is proportional to the complexity, hence to the number of model variables. However, in a spectral model we will account for the fact that the time taken by Legendre transforms increases more rapidly than linearly with the number of variables.

In addition to solving the primitive equations, NWP models also need parametrisation to describe sub-grid scale processes, surface properties, radiation and phase transitions. These processes are collectively called "physics", as opposed to the primitive equations, that are called the "dynamics". We assume that the computation time in physics is proportional to the number of model variables, like we did in dynamics (except in the Legendre transforms).

This proportionality is almost universally observed in NWP models, as long as the model complexity is only changed by varying the number of model variables. However, if the model is modified to use a different solution method or a different parametrisation scheme, this may result in a change in computation time even if the number of model variables is kept constant. As argued above, this is unlikely to happen for dynamics. In physics, on the other hand, we may expect the development of more expensive schemes; this development is counter-acted by the reduced need for the parametrisation of sub-grid scale processes if the resolution increases. Our assumption that the computation time of physics is proportional to the number of model variables is therefore more or less equivalent to the assumption that these two counter-acting effects balance out.

In current models, physics requires more computation time per time step than dynamics. The dynamics may put a constraint on the time step due to stability of the integration scheme; the time step for the physics, on the other hand, is solely limited by accuracy. It is therefore efficient to run the dynamics with the largest time step allowed by stability and accuracy, and constrain the physics time step only by accuracy. Hence, we will assume that the model uses two different time steps. We will also assume that the most complicated current models (sisLsp) run at the maximum time step allowed by accuracy, both for physics and dynamics. So in this paper, we take this accuracy constrained time step for physics for all kinds of NWP models; but for dynamics we will use a smaller time step if the model formulation requires so for stability.

The number of variables is equal to the number of grid points times the (average) number of model variables per grid point. The number of grid points is the number of horizontal grid points times the (average) number of points in the vertical. The number of grid points in the horizontal determines the horizontal resolution. In particular the horizontal resolution determines the size of the weather systems that can be described by the model. That size in its turn determines the life time of those systems: smaller systems live shorter. We will hence assume that the size of the physics time step is proportional to the horizontal distance between grid points. In explicit models (*e.g.,* exEugp) the dynamics time step is also proportional to that distance, for stability reasons. In the more complicated models (in particular sisLsp) the dynamics time step also obeys that proportionality, but here for accuracy reasons.

If the horizontal model domain is a square, for horizontal homogeneity we would require that the number of grid points is equal in the two horizontal directions

$$N_y = N_x \ . \tag{2}$$

On the globe we have 360 degrees of longitudes, and only 180 degrees of latitudes. On such a domain, homogeneity would be expressed more like

$$N_y = N_x/2 \ . \tag{3}$$

However, whatever the model domain is, homogeneity will always result in the requirement that the number of grid points in one direction is proportional to that number in the other direction, and hence both are proportional to the square root of the number of grid points in the horizontal.

If we further assume that the model domain will remain constant (it is now global in many applications and hence cannot increase anymore) then the number of grid points is inversely proportional to the horizontal resolution.

From past developments we will be led to the assumption that the (average) number of variables per horizontal grid point (which is equal to the number of grid points in the vertical times the average number of model variables per grid point) is proportional to the number of grid points in one horizontal direction.

Combining all these we obtain that the complexity of an NWP model can be measured by its number of grid points in one of the three directions. For this measure, we will use the x-direction, because it is directly related to the model resolution. The total number of model variables will be proportional to the third power of this.

In operational weather forecasting, usually a fixed time window is available within which the model should complete. Usually this time window is in the order of one hour. We will assume that this allowable time window will not change in the future. So we arrive at the following criterion to judge the value of a model: if we have a variety of NWP models, we will prefer the model that allows the largest number of grid points to be processed on the available computer within one hour.

In summary, we make the following main assumptions on the NWP model:

1. The value of a model is set by its resolution, *i.e.,* the number of grid points in one horizontal direction that can be evaluated within one hour. The total number of model variables is proportional to the third power of this.

2. Computation time is proportional to the number of model variables. Legendre transforms, as used in global spectral models, are more expensive.

3. The time step of the physics is inversely proportional to the number of grid points in one horizontal direction. Whatever model we use, it is equal to the time step of the most stable model currently known, *i.e.,* sisLsp.

4. The time step of the dynamics is also inversely proportional to that number of grid points, but for Eulerian or explicit models stability requires the use of a smaller time step than for physics.

3.2 Computer architecture

Throughout the core of the paper we will assume that the currently used computer architecture is valid for future computer configurations. So we will assume that in time we have a growing number of processors, where each processor has a growing performance. This is the "business as usual" scenario.

The processors are connected by a fast interconnecting network. For this network we will make optimistic assumptions. The communication between a pair of nearest neighbours is assumed to proceed completely independently of and in parallel to the communication between any other (but disjunct) pair of nearest neighbours in the network. This may perhaps be achieved with a torus-like network topology. But we will also assume that global communications, so between non-nearest neighbours, has the same property. This would require a very high performance switched network.

The latency of the network is assumed to consist of a component due to start-up time of the messages, and a component due to the limited bandwidth. Both are assumed to decrease in the future, but the start-up time is assumed to be limited by the speed of light. For that, we will have to make assumptions on the physical size of the computer (the "footprint") and on the number of messages sent per model time step. We will let the footprint grow with the number of processors, up to a certain maximum. At this stage, we can distinguish between conventional and Grid computers: the latter have a larger footprint per processor.

If an NWP model needs nearest-neighbour communications, the longest physical distance between two nearest neighbours limits the communication performance over the whole network: even if other nearest neighbour

pairs may complete their communications faster, they still will have to synchronise with the slowest communication in the network. This synchronisation is needed every dynamics time step. Similarly, if an NWP needs global communication, it is the longest distance between any two processors that will limit the overall network performance.

The contemporary hardware tendency into multi-core processing does not immediately fit into these assumptions. Yet, if we consider a multi-core processor as merely a faster single-core processor, the results of this paper can straightforwardly be applied to multi-core processing. We will come back to this in Section 8 The main issue with multi-core processing as far as it may influence our results is that we do not have any idea how fast multi-core processing will develop, because we do not have any data yet for extrapolations into the future. So if multi-core processing indeed will set off, our extrapolations in time may have to be changed to a different time scale.

The main assumptions on computer hardware developments are:

1. Per-processor performance will continue to grow.

2. The number of processors will continue to grow.

3. Per-message communication time is a sum of start-up time and a time proportional to the inverse of bandwidth.

4. Start-up time will continue to decrease, but it will be limited by the speed of light.

5. Bandwidth will continue to grow, faster than the decrease of the start-up time.

6. The network will be fast enough to support fully parallel communication between any two disjunct pairs of processors.

7. The worst communication time in a network will determine the performance of the entire network through synchronisation. In view of the previous assumption, this means that the longest distance between nearest neighbours in the network will limit the local communication speed, whereas the longest distance between any two processors in the network will limit the global communication speed.

3.3   Projecting the NWP model layout onto the computer architecture

For our model, we are also going to need an assumed projection of the NWP model topology onto the network topology. For this, we will assume the straightforward horizontal domain decomposition paradigm. Each processor will process a rectangular sub-domain of the globe. The neighbouring rectangles will be processed by neighbouring processors. Strictly speaking, this topology requires that the domain of the NWP model is rectangular. However, the general concept can also be used for *e.g.,* a global model, by dividing the globe into more or less equally sized patches. The borders between the patches become more complicated than in a purely rectangular domain, but this does not change communication patterns substantially; local communications remain local, although an intermediate processor may be involved if the network connections only allow nearest-neighbour communications. We will base our analysis on the simple rectangular domain decomposition (Figure (1)).

At some stage in the future, the number of processors may have become bigger than the number of grid points in the horizontal. The standard domain decomposition paradigm then breaks down. The amount of communications in the vertical during physics is so big that it is probably not wise to venture then into a three-dimensional domain decomposition. Hence, in our model we will simply assume that if the number of processors exceeds the number of grid points in the horizontal, the size of the computer will be reduced to have one grid point per processor. Note that this may require an iterative process: if number of processors and hence the size of the computer is reduced, communication time will diminish, allowing more grid points to be processed within one hour, and thus allowing more processors in the two-dimensional domain decomposition.

We note that three-dimensional domain decomposition may become feasible when communication times are very small. We expect that to be the case within a multi-core processor, where communication is by common memory. Yet, if -as before- we consider a multi-core processor merely as a fast single processor, our horizontal domain decomposition assumption will remain valid: how the single processor achieves its speed is not our concern here. Possibly internally the multi-core processor indeed is programmed for a three-dimensional domain decomposition, but the array of multi-core processors will still have to be configured in a horizontal decomposition, to avoid a multitude of vertical communications between the processors.

If the NWP model needs calculations in non-physical, *e.g.,* spectral, space, it is assumed that after the required global communications, the calculations can be done in parallel, without further communication. So, if the model needs a Fourier transform for each variable, there will be global communications before and possibly after the Fourier transform; the transform itself proceeds on a single processor, one for each model variable. In this way, also the calculations in non-physical space are fully parallel.

We will assume that global two-dimensional communication is implemented as a succession of two one-dimensional communications. Later on we will argue that in NWP models this will lead to faster communications than arbitrary communication between any pair of processors, primarily because the number of messages can be much smaller.

The main assumptions on the model and computer topologies are:

1. Parallelisation of the calculations in physical space is through horizontal domain decomposition.

2. Hence, communications that are local in physical space, are also local in the computer network. They are constrained by the largest distance between any two nearest neighbour processors.

3. Also a consequence of the horizontal domain decomposition is that calculations in non-physical space, *i.e.,* those requiring data transposition, require global communications in the computer network. They are constrained by the largest distance between any two processors in any of the two horizontal directions.

4. Even though those calculations require global data, they are also performed in parallel.

5. The number of processors will never be in excess of the number of grid points in the horizontal.

4   A simple model for total execution time

Total execution time T_e of a (numerical weather prediction) application is modelled as the sum of a computation time T_p and a communication time T_m, with a correction for possible overlap between communications and computations $T_{p \cdot m}$:

$$T_e = T_p + T_m - T_{p \cdot m} \ . \tag{4}$$

The communication time T_m is assumed a sum of a time due to latency per message and a time that is proportional to the number of data to be transferred.

In this section, we will attempt to express the two terms T_p and T_m in the parameters of the application and of the hardware. The overlap, $T_{p \cdot m}$, will be smaller than either. In our baseline configuration, we will assume that it is negligible, but to test sensitivity, we will also consider a configuration with maximum overlap.

The application and hardware parameters are in particular: the number of model variables, the number of processors, the computational power per processor and network parameters.

The first assumption is that the computation time is proportional to the number of model variables and inversely proportional to the number of processors. This implies that computations are assumed perfectly scalable, which is reasonable because computations are embarrassingly parallel. This results in

$$T_p = (T_{p \cdot d} N_{t \cdot d} + T_{p \cdot p} N_{t \cdot p}) \left\lceil \frac{N_x}{p_x} \right\rceil \left\lceil \frac{N_y}{p_y} \right\rceil N_{v \cdot z} \ . \tag{5}$$

In here, $T_{p \cdot d}$ and $T_{p \cdot p}$ are proportionality factors, measuring the time per time step and variable spent in dynamics and physics, *resp.,* ; $N_{t \cdot d}$ and $N_{t \cdot p}$ are the number of time steps taken by dynamics and physics, *resp.,* ; N_x is the number of grid points in the x-direction, p_x the number of processors in that direction; similarly for the y-direction variables. We use the notation $\lceil \frac{a}{b} \rceil$ for the ceiling of, *i.e.,* the integer nearest to and not smaller than, a/b. The total number of variables in the vertical is denoted by $N_{v \cdot z}$. This usually is close to the number of grid points in the vertical multiplied by the number of variables per grid point, but it may deviate slightly; *e.g.,* if a model has surface pressure, this would count as an additional variable to $N_{v \cdot z}$.

Obviously, the total number of model variables at a time step is given by

$$N_v = N_x N_y N_{v \cdot z} \tag{6}$$

and the total number of processors is

$$N_p = p_x p_y \ . \tag{7}$$

In a spectral model, the Legendre transforms are usually feared to require a computation time that increases faster than the number of model variables. Côté and Staniforth [7] mention that the costs are proportional to N_x^3, faster by a factor of N_x than the proportionality to $N_x \times N_y$ as assumed in Equation (5). However, there are indications that an algorithm can be developed to allow costs to increase as $N_x^2 \times \log^2 N_x$ [29]. We will use this more optimistic view, and hence assume that for a spectral model the part of $T_{p \cdot d}$ that is due to the Legendre transforms, is proportional to $\log^2 N_x$.

The communication time requires quite a few more assumptions in order to express it in terms of model and hardware configuration parameters. An NWP model performs a number of local communications and a number of global communications at every dynamics time step:

$$T_m = T_{loc} + T_{glob} \ , \tag{8}$$

where both T_{loc} and T_{glob} will be proportional to $N_{t \cdot d}$. Both T_{loc} and T_{glob} will be modelled as the sum of a constant start-up time and a term proportional to the number of bytes transferred. The inverse of the proportionality factor in the second term is usually referred to as the bandwidth. This simple model is commonly used [24], although far more complicated models exists (*e.g.,* [20]).

We will assume that the local communications scale perfectly with the number of processors. This is a fair assumption in a network where each processor is connected with its four nearest neighbours. In such a network, all processors may communicate with their nearest-neighbours in parallel. In practice, even in such a network one often observes an increase in communication time with an increased number of processors, but we will neglect that effect here. The amount of data that is communicated per time step is proportional to the length of the border between the domains of neighbouring processors, *e.g.,* in the x direction $\lceil \frac{N_y}{p_y} \rceil N_{v \cdot z}$.

In a semi-Lagrangian model, a processor needs information from a zone around its own domain. This zone is called a "halo". The width of a halo zone, W_{halo}, is typically 7 to 10 grid points. If p_x becomes so large that $\lfloor \frac{N_x}{p_x} \rfloor$ decreases below the width of the halo zone, nearest neighbour communications are not sufficient anymore. (We used the notation $\lfloor \frac{a}{b} \rfloor$ to denote the floor of a/b.) Instead, information will be needed from processors as far away as

$$p_{x,halo} = \left\lceil \frac{W_{halo}}{\left\lfloor \frac{N_x}{p_x} \right\rfloor} \right\rceil \tag{9}$$

and a similar expression in the y direction. When communication further away than 1 processor is needed, usually also the intermediate processors will have to provide information. So $p_{x,halo}$ in Equation (9) is also measure for the number of communications in the x direction. In a nearest neighbour network communications between processors more than 1 processor apart will be more costly than between neighbouring processors. We will ignore this effect. By performing communications first in one horizontal direction and then in the other,

the total communication time becomes

$$T_{loc} = N_{t \cdot d} p_{x,halo}(L_{loc} + B_{loc} \left\lceil \frac{N_y}{p_y} \right\rceil N_{v \cdot z}) + (\dots) \,, \tag{10}$$

where L_{loc} is proportional to the latency and B_{loc} to the inverse of bandwidth for local communications. The $(\dots)$ denote the same term but with x and y subscripts swapped. We note that this expression, complicated by the occurrence of the factor $p_{x,halo}$, can be used for Eulerian models as well, provided we take the halo width small (1, in some formulations perhaps 2), resulting in $p_{x,halo} = 1$.

For the global communications in Equation (8), we will assume the following particular communication pattern: In a first step, within each of the p_y slices in the y-direction all data are transposed. Hence, each of the p_x processors in one such slice exchanges messages with each of the other $p_x - 1$ processors in the same slice. In a nearest neighbour network topology, it is reasonable to assume that the slices do not hinder each other, in other words, the p_y slices operate fully in parallel. Within a slice, on the other hand, messages may have to travel over many network connections, resulting in possible contention and congestion delays. *However, here we will make the most optimistic assumption possible, namely that such delays do not occur.*

Because each processor will have to handle $p_x - 1$ messages, communication delay will be proportional to this number. The amount of data to be transferred per message is the number of variables held on each processor, $\left\lceil \frac{N_x}{p_x} \right\rceil \left\lceil \frac{N_y}{p_y} \right\rceil N_{v \cdot z}$.

In a second step, within each of the p_x slices, now in the z-direction, the p_y processors communicate with each other. The number of communications per processor is $p_y - 1$. Again under the most optimistic assumption, communication delay will be proportional to this. The number of data in each message is $\left\lceil \frac{N_{v \cdot z}}{p_x} \right\rceil \left\lceil \frac{N_y}{p_y} \right\rceil N_x$.

The total time for global communications can now be expressed as

$$
\begin{aligned}
T_{glob} \quad = \quad & N_{t \cdot d}(p_x - 1)(L_{glob} + B_{glob} \left\lceil \frac{N_x}{p_x} \right\rceil \left\lceil \frac{N_y}{p_y} \right\rceil N_{v \cdot z}) + \\
& N_{t \cdot d}(p_y - 1)(L_{glob} + B_{glob} \left\lceil \frac{N_{v \cdot z}}{p_x} \right\rceil \left\lceil \frac{N_y}{p_y} \right\rceil N_x) \,,
\end{aligned}
\tag{11}
$$

where L_{glob} and B_{glob} are proportionality factors for start-up time and inverse of bandwidth, *resp.,* for global communications.

The above communication model, where communications are first performed in one direction and then in the other, is not the only one conceivable. For example, one might also organise the communications such that each processor sends the data it holds directly to the processor where they are required. This would result in many, $(p_x p_y - 1)$, but short messages. In a network where delays due to start-up time are much smaller than those due to bandwidth, such a communication strategy may be preferable, in particular because the coding is easier and it avoids the memory to memory copies for the aggregation and decomposition of the exchange buffers. Indeed, the first massively parallel implementation of HIRLAM followed this pattern, because its target machine, T3E, has a very good start-up time to bandwidth ratio. However, network architecture developments since then show a faster growth of bandwidth than decrease of start-up time [24]. Consequently, HIRLAM has been reprogrammed [3]. Therefore, this communication scenario is not considered viable anymore.

5   Approximate scaling behaviour

5.1   Introduction

The intent of this section is to develop a feeling for the behaviour of the several terms that add up to the total execution time, when the total number of processors is increased. So we will investigate how, *e.g.,* the

number of grid points scales with the number of processors. We will do this based on the currently common configurations.

A characteristic of NWP implementations is that with increasing processing resources the total execution time remains constant. In other words, where in the course of time more computing power becomes available, the additional resources will not be used to decrease elapsed time, but to solve a more complicated set of model equations. In practice, this usually means increasing the total number of model variables, N_v. Hence, the scaling behaviour of the NWP system will be studied based on the assumptions that total execution time will remain constant and that increased computing power will be used to increase N_v.

In current configurations, total execution time is dominated by the computation time: in Equation (4), $T_p >> T_m$ and, because $T_{p \cdot m} \leq T_m$, also $T_p >> T_{p \cdot m}$. In fact, for this reason not much coding effort has been put so far into optimising the overlap. So in this section we will neglect the communication/computation overlap $T_{p \cdot m}$.

5.2   Relation between number of grid points and number of processors

Current implementations have $N_x >> p_x$ and $N_y >> p_y$. The ceiling operator in Equation (5) may hence be dropped. The equation thus becomes to good approximation

$$T_p = (T_{p \cdot d} N_{t \cdot d} + T_{p \cdot p} N_{t \cdot p}) N_x N_y N_{v \cdot z} p_x p_y \ . \tag{12}$$

For isentropy in the horizontal, we will assume that N_x and N_y are proportional, which we will write as $N_x \sim N_y$. Although the number of variables per horizontal grid point can be chosen fairly independently of the horizontal number of grid points, in practice we see that increases in horizontal resolution are more or less followed suit by increased vertical resolution and/or increased number of model variables.
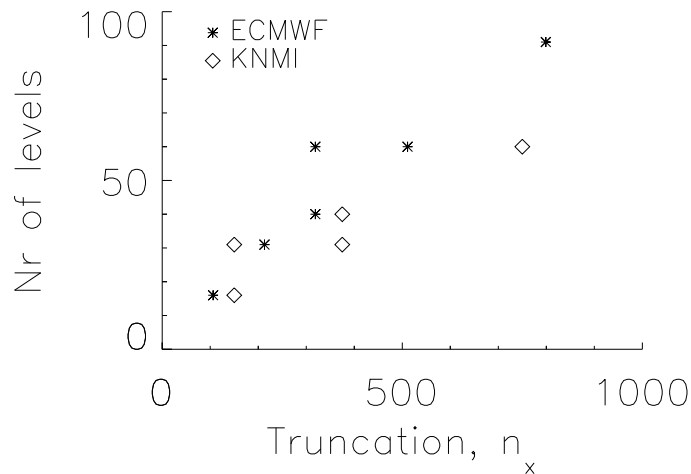

Figure 2: the number of levels in operational model configurations: at ECMWF, against the triangular truncation; at KNMI against the number of grid points in the x-direction.

As an example, Figure (2) shows that the number of levels in the ECMWF model [33] in the course of time is reasonably proportional to the spectral truncation, which is a measure for N_x and N_y. A similar proportionality is observed between the number of levels and the number of grid points in the x-direction for an operational implementation of a grid point model, namely HIRLAM [28] at KNMI. An occasional outlier in Figure (2) merely supports the statement that the number of levels is not tightly coupled to N_x. Yet, we will assume $N_x \sim N_{v \cdot z}$, where we are aware that the proportionality factor is bigger than as suggested by Figure (2), because in the course of time not only the number of levels has increased, but also the number of variables.

We will assume that current models, with the semi-implicit and semi-Lagrangian formulations, operate with the maximum time step that can be used to keep time truncation errors tolerable. Consequently, the physics package has to be invoked every time step, *i.e.,* $N_{t \cdot d} = N_{t \cdot p}$. Assuming that physical phenomena have time scales that are proportional to their horizontal scale, another consequence is that the time resolution should be proportional to the horizontal resolution, $N_{t \cdot d} \sim N_x$.

As we see in Equation (11), the number of communications increases more or less proportional to the sum of p_x and p_y. At a fixed number of processors, $p_x p_y$, a minimum in communication cost is achieved if p_x and p_y increase in proportion, $p_x \sim p_y$.

Collecting all proportionalities, and using that T_p does not change with increasing number of processors, Equation (12) leads to

$$N_x \sim \sqrt{p_x} \ . \tag{13}$$

Hence, the total number of horizontal grid points assigned to a processor, N_x/p_x, decreases with increasing number of processors. The resulting savings of computation time are absorbed by the increased number of variables $N_{v \cdot z}$ and by the larger number of time steps needed.

The proportionality Equation (13) results in the following relation between the total number of variables and the total number of processors

$$N_v \sim \sqrt[4]{N_p}^3 \ , \tag{14}$$

so the total number of variables increases slightly slower than the number of processors. The number of variables held on one processor, $\frac{N_x}{p_x} \frac{N_y}{p_y} N_{v \cdot z}$, decreases like $1/\sqrt{p_x}$. This is a measure for the number of memory accesses per time step. The number of time steps increases like $\sqrt{p_x}$. Because the total execution time is fixed, this implies that the time slot allowed for one time step is proportional to $1/\sqrt{p_x}$, and hence the memory access rate is independent of p_x.

When the number of processors becomes very large, the number of grid points per processor would eventually drop below 1. At that stage, the horizontal domain decomposition paradigm would break down, and one would have to consider other parallelisation strategies, like vertical domain decomposition or parallelising over processes. However, in this section, and in fact throughout this whole paper, we will refrain from entering this domain.

We will use proportionality Equation (13) now to get a first impression of the effect on communication time when model resolution and number of processors are increased.

5.3   Local communications

Let us first look at the local communication time, T_{loc}. In Equation (10) the complicating factor is $p_{x,halo}$. In current implementations, $\frac{N_x}{p_x} >> W_{halo}$, so we will put $p_{x,halo}$ to 1. The proportionality Equation (13) then shows that T_{loc} increases as $\sqrt{p_x}$, *i.e.,* as $\sqrt[4]{N_p}$, with increasing p_x, but the distribution of the delays over latency and bandwidth does not change. We keep in mind that when p_x becomes large, and thus, according to Equation (13), N_x/p_x becomes small, say less than 10, this proportionality will no longer be valid for semi-Lagrangian models.

5.4   Global communications

In the expression for global communication time, Equation (11), we encounter the fraction $N_{v \cdot z}/p_x$. Typical current values for it are 20 to 50, smaller than N_x/p_x, but still large enough to be able to neglect the effect of the need to round this fraction upward to the nearest integer. The proportionality Equation (13) then tells us that the losses due to bandwidth are proportional to p_x and those due to start-up to $\sqrt{p_x}^3$.

5.5 Discussion

The discussion above has been based on the assumptions that total execution time remains constant and that computation time dominates communication time. We then assumed the computation time to be independent of the number of processors, and derived Equation (13). However, the scaling results show then that local communication overhead increases as $\sqrt{p_x}$, and global communication overhead even as $\sqrt{p_x}^3$ with increasing p_x. Hence, at larger values of p_x the assumptions will no longer be valid.

Another silent assumption we made was that the proportionality factors for computations ($T_{p \cdot d}$ and $T_{p \cdot p}$) and for communications (L_{loc}, B_{loc}, L_{glob} and B_{glob}) do not depend on the number of processors. In practice, however, we see that in the course of time not only the number of processors increases, but also their computational power and the network bandwidth and latency improve. So in practice there is an apparent relation between the number of processors and the proportionality factors. In the following section, we turn to address this issue.

We repeat the conclusion in Subsection 5.2 that the memory access rate is independent of the number of processors, and hence does not need attention in this paper.

6   Empirical relations

6.1   Introduction

In Section 4 we introduced a number of proportionality factors that themselves are related to hardware performance parameters. Then, in Section 5, we concluded that it is not just sufficient to establish a relation between number of model variables and number of processors, but that we also need a relation between the number of processors and those performance parameters. Now, logically, there is no reason for such relation to exist. Yet, in practice, in the past we observed that both increased over time, which might be interpreted as an empirical relation between the two sets of parameters. The intent of this section is to use the past empirical growth rates of the parameters to find extrapolations for their growth in the future.

Empirical evidence for almost all parameters points to an exponential growth. This is expressed by

$$P_x \mathcal{N}_x = \pi_x \alpha_x^{-y} \ . \tag{15}$$

In here, P_x stands for any performance parameter we introduced before (listed in the first column of Table (6.1)), and x for any subscript we introduced before. We will use the notation of a lower case Greek letter to reflect the value of the parameter in the reference year (we took 2006) and y is the number of years since the reference year, negative in the era before the reference year. In Subsection 6.2 we will derive values for all relevant parameters α_x.

In Equation (15) we added a normalisation factor, $\mathcal{N}_x$. This factor serves to take out the number of grid points, *etc.*, in the reference year from π_x. The effect is that π_x gives the time spent in a certain process in the reference year. As an example, Equation (5) shows that

$$\mathcal{N}_{t \cdot d} = N_{t \cdot d} \left\lceil \frac{N_x}{p_x} \right\rceil \left\lceil \frac{N_y}{p_y} \right\rceil N_{v \cdot z} \ , \tag{16}$$

where the right hand side must be evaluated in the reference year.

If communication times decrease exponentially with time, they will eventually tend to zero. But we should expect that they will not decrease below the delays due to the finiteness of the speed of light. We will discuss this issue in Subsection 6.4.

In Subsection 6.5, finally, we will find values for the variables π_x in Equation (15), based on data for the ECMWF model, provided by Salmond [35].

Table (6.1) summarises the results.

Table 1: Parameters describing computer performance over time, as per Equation (15). Numbers α give decrease per year. The reference value is the value in the reference year, 2006, derived from [35].

	Eq.	description	yearly decay	reference value (s)
$T_{p\cdot d}$	5	dynamics comps time	$\alpha_{p\cdot d}$=1.2	$\tau_{p\cdot d}$=1210
$T_{p\cdot p}$	5	physics comps time	$\alpha_{p\cdot p}$=1.2	$\tau_{p\cdot p}$=1600
L_{loc}	10	local comms, start-up	$\alpha_{L\cdot loc}$=1.26	λ_{loc}=9
B_{loc}	10	local comms, $1/BW$	$\alpha_{B\cdot loc}$=1.58	β_{loc}=171
L_{glob}	11	global comms, start-up	$\alpha_{L\cdot glob}$=1.26	λ_{glob}=18
B_{glob}	11	global comms, $1/BW$	$\alpha_{B\cdot glob}$=1.58	β_{glob}=342

6.2   Performance parameters over time

In this subsection we will motivate our choices for the yearly decay rates, α_x, as shown in Table (6.1).

We will first focus on compute performance. Hennessy and Patterson [24] show in their Figure 1 in Chapter 1 that recently compute performance has been seen to increase by 20% per year. We assume this figure for the yearly compute performance growth also for the future, and express this in the yearly decrease of compute time for both physics and dynamics ($\alpha_{p\cdot d} = \alpha_{p\cdot p} = 1.2$). In principle, the compute time includes memory access time. Memory bandwidth grows much more slowly than compute performance [24], but we do not have to worry about that, because the scale analysis at the end of Subsection 5.2 has shown that the required memory bandwidth does not change with the number of processors. Memory access is negligible nowadays, and hence will remain so in the future.

There is little literature on the the development of the message passing start-up times. In Figure (6.2), which has been derived from Culler and Pal Singh's [8] Table 7.1, and from data provided in [25], we observe an initial decay of over an order of magnitude in 10 years, but apparently over the past decade the improvement was more in the order of a factor of 10, corresponding to a yearly decrease of 26%. We assume that this can be used for both the local and global start-up times, L_{loc} and L_{glob}.

Culler and Pal Singh [8], and more recently, Hennessy and Patterson [24] state that bandwidth improves faster than start-up time. The latter authors give as a "rule of thumb" that bandwidth improves at least twice as fast. We implement this rule of thumb by choosing the bandwidth growth rate equal to the square of the start-up time decay rate. Again, we use this for both the local and global inverse of bandwidth, B_{loc} and B_{glob}.

6.3   Number of processors over time

With the decrease of per chip production costs and per chip power consumption computer installations are being delivered with an increasing number of processors.

The Dongarra Linpack [12] benchmarks provide us with an impression of the growth of the number of processors, in time; see Figure (4).

The number of processors in the top ranking machines shows an initial steep increase between 1990 and 1994. Then follows a relatively flat area until 2004. In this episode, compute performance increased mainly due to better per-processor performance. In 2004 and 2005 we observe a steep increase again. These two points in the Figure are for machines in a defence institution. It is rather dangerous to extrapolate a curve of this
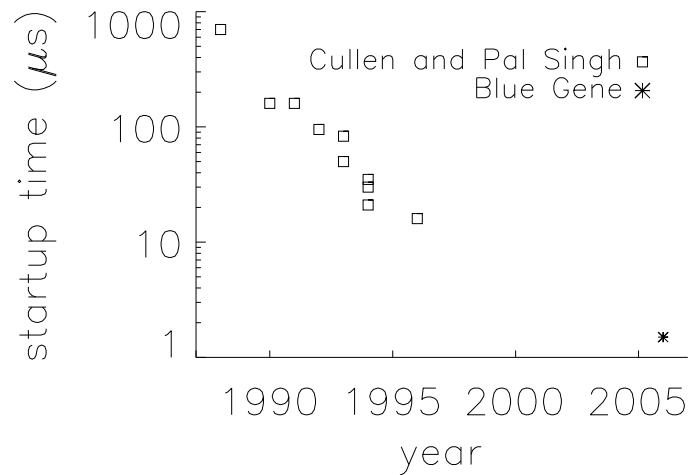
Figure 3: Communication start-up time between 1988 and 1996, according to Table 7.1 of Culler and Pal Singh [8] and IBM Blue Gene in 2006 [25]
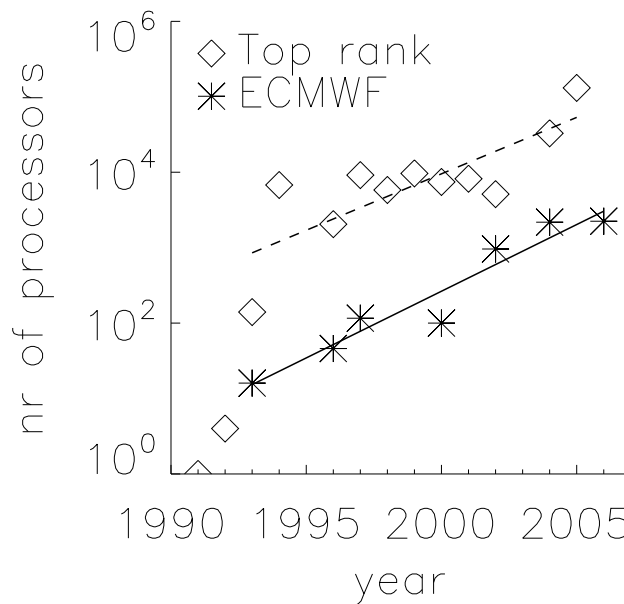

Figure 4: Number of processors in large installations, according to the Dongarra Linpack [12] benchmarks, for top ranking machines and at ECMWF. *The full line is the least-square fit to the* ECMWF *points, the dashed line to the top ranking points, but excluding the 1990 and 1991 points.*

nature. On the other hand, the number of processors at ECMWF shows a more regular increase, with perhaps an exception for the year 2000. It is probably more appropriate to consider the situation at ECMWF than that at a defence organisation, because it reflects better yearly development of the budgets available for numerical weather prediction; fortunately, it is also less of an adventure to fit the observations with a straight line (the correlation coefficient is 0.96). Such a fit then shows that at ECMWF the number of processors increases tenfold in 6 years

$$N_p = \nu_p \alpha_p^y \,, \tag{17}$$

with $\alpha_p = 1.50$. This, by the way, is not far from the slope of the best fit to the top ranking machines data points, if we drop the 1 and 4 processor points in 1990 and 1991. So, the number of processors of the machines at ECMWF follows that of the top machine of the world, but, as seen in Figure (4), with a time lag of 9 years.

6.4   Speed of light, extent of the computer

Start-up times of communications in computers will be at least as big as the time it takes the signal to travel between the processors. In computers of current design this time is determined by the size of the computer and the speed of light. For reasons of causality, it is extremely unlikely that it will ever be possible to exchange information at a higher speed [4].

To quantify the limitations posed by the finite speed of light we have to make assumptions on the physical size of the computer. Our results will, fortunately, not depend strongly on the choices we are going to make below for some fairly arbitrary numbers.

For local communications, we will call the biggest distance between any pair of processors that hold data of neighbouring grid points d. We will usually optimistically choose d equal to 0.1 m for a classical computer. For a Grid computer, we will take d equal to 30 km. For global communications, we assume that all processors are separated by d, with a maximum of at most 30 processors stacked in the vertical. The number of processors thus determines the horizontal size of the computer, but we do not allow it to grow over 100 m for a classical, and 3000 km for a Grid computer.

If the domain held by a processor is too small to hold the full halo for its neighbour, the communication distance will become larger, so we take it proportional to $p_{x,halo}$ (see Equation (9)). The limits posed by the speed of light will be proportional to the number of messages sent or received by a processor. We modelled that number already in proportionality to the number of processors and time steps (see Equation (11)), but it will also be proportional to the number of messages that the communication schedule needs per time step. We assume that an explicit Eulerian grid point model can be coded to require very few communications per time step (we chose 2). A semi-Lagrangian scheme, however, will need more communications, because trajectories are calculated with an iterative scheme. We assume 20 local messages for a semi-Lagrangian model.

In a spectral model or in a semi-implicit model we assume that we need 20 messages per time step for global communications. This fairly large number is based on Figures in [34].

In general, we will assume that the start-up times (L_{loc}) is the bigger of the value following the exponential decay of Table (6.1) and the value c/d, derived from the finite speed of light c, and similarly for global messages. However, in a Grid computer we will use the sum of those two numbers, rather than the bigger one, which choice will also serve to assess the sensitivity to this formulation.

6.5   Relative contributions of computations and communications

In order to complete the model for total forecast model costs we now have to establish values for the different proportionality factors ($\tau_{p \cdot d}$, $\tau_{p \cdot p}$, λ_{loc}, β_{loc}, λ_{glob} and β_{glob}) in current operational models.

We chose to derive these quantities from data provided by Salmond [35] on the ECMWF model. This model is semi-implicit, semi-Lagrangian and spectral. Salmond's data are summarised in Table (6.5).

We combined the processes that are executed every dynamics time step into the dynamics computation time. The ECMWF model is spectral, so we will take into account that the Legendre transformations (at 210 s) will grow faster than the other components, when the number of grid points increases.

We take the time spent in radiation proportional to the number of physics time steps, even though radiation is not executed every time step.

The semi-Lagrangian communications, SL comms, are taken to be the local communications time, and the Transpositions form the global communications time. Currently, the messages in the ECMWF model are large and hence start-up time is negligible as compared to delays due to limited bandwidth [34], [36]. However, we will have to make a choice; in our baseline configuration, we will assume that start-up takes 5% of communication time. In one sensitivity run, we will try a choice of 1%.

Table 2: Time spent in several processes in the ECMWF *model, in the reference year 2006. The numbers have been derived from the pie charts by [35], for T799, scaled such that the total execution time of the model is 1 hour.*

process	time *cf.* [35](s)	entered into Table (6.1)		
Dynamics	200			
Spectral	100			
Semi-Lagr	470			
WAM	200			
Fourier tr	30			
Legendre tr	210			
Total dynamics computations		$\tau_{p\cdot d}$=1210 s		
out of which Legendre tr grow like $\log^2 n$				
Physics	1320			
Radiation	280			
Total physics computations		$\tau_{p\cdot p}$=1600 s		
SL comms	180	5%	λ_{loc}=9 s	
		95%	β_{loc}=171 s	
Transpositions	360	5%	λ_{glob}=18 s	
		95%	β_{glob}=342 s	
Barrier	220			

All our assumptions about communication patterns were optimistic, in the sense that no time is lost in synchronisation. The data in [35], however, show that 210 s, *i.e.,* 6%, is lost in Barrier. We will simply ignore this contribution, being aware that our model will have many other inaccuracies of much larger impact.

We now choose the different proportionality factors ($\tau_{p\cdot d}$, $\tau_{p\cdot p}$, λ_{loc}, β_{loc}, λ_{glob} and β_{glob}) such that in 2006 the corresponding contributions to overall elapsed time equal those reported in [35]. For this, we used that the data in [35] are valid for a T799 truncation. The forecast length was 10 days, with a time step of 15 minutes. Hence, there were 960 time steps, both for physics and dynamics.

6.6 Summary

In this section we introduced a number of proportionality factors, denoted by Greek lowercase characters. These factors aggregate the cost of computation and communication per model time step and per unit of model size, where model size is expressed in number of grid points in one of the horizontal directions. The decreases in those costs due to hardware improvements over time have been eliminated from those proportionality factors by assumed exponential relationships. The growth factors in those relationships have been estimated from the literature. They show a 20% yearly improvement for the computation time ($\alpha_{p\cdot d} = \alpha_{p\cdot p} = 1.2$), a 26% yearly improvement of communication start-up time, a 58% yearly improvement of bandwidth and a 50% yearly increase of number of processors. We made the proviso that communication start-up time must be bounded by the speed of light, and described how we will take account of this.

The different proportionality factors are more or less independent of the choice of model, with the exception of the dynamics calculation time: it is bigger for semi-Lagrangian than for Eulerian models, due to the trajectory calculations; it is bigger for spectral than for grid point models, due to the spectral transformations; and it is -albeit only slightly- bigger for semi-implicit than for explicit models, due to the solution of the Helmholtz equation. The global communication costs depend on model formulation in the sense that in particular an explicit model does not need global communication. Note that the physics computations are independent of model formulation.

To quantify the relative contributions of computation time (dynamics and physics) and of communication time (local and global), we used data for the ECMWF model in 2006, provided in [35]. We had to assume some distribution of communication times over start-up and bandwidth-related. We chose to use that 5% of communication time is spent in message start-up.

7   Results

7.1   Introduction

In this section, we are going to present figures with two or more panels. The panels on the left display the maximum number of grid points that can be processed within one hour on a computer as was or will be available as a function of time. The panels on the right show the relative contributions of computation and communication to total elapsed time, also as a function of time. The compute and communication performance of the computer is extrapolated from the situation in 2006 using the empirical relations from Section 6.

For the purpose of this presentation, we refer to N_x as the number of grid points. Of course, the total number of grid points is then $N_x \times N_y \times N_z$, but because all these quantities are proportional to each other, it suffices to present and discuss N_x on its own.

In general, when several models are available, we would prefer the model that achieves the highest resolution, *i.e.,* the highest number of grid points on a certain computer. In this sense, the number of grid points as displayed in the left panels in the Figures is a measure of model performance. For a spectral model, we assume that the number of grid points that is a measure of model performance is twice the spectral truncation, so $N_x = N_y = 2 \times N_T$ [21]. We will call this the "equivalent number of grid points for a spectral model".

In a grid point model, atmospheric waves need more than 2 grid points to be representable. We will assume that 3 grid points in each horizontal direction suffice, and thus arrive at the "equivalent number of grid points for a grid point model", which is $2/3$ of the real number in each horizontal direction, and $4/9$ in the horizontal plane.

Because 3 grid points represent waves very crudely, we know that by using this definition the accuracy of the dynamics is less than that in a spectral model at the same number of equivalent grid points, but the accuracy of the physics will be higher, because we have more grid points to represent things like land/sea distribution. We assume that these two effects result in the equivalent number of grid points being a direct measure of "desirability" of the model: the higher the equivalent number of grid points, the higher its accuracy.

In the right panels, the time for computations is split into physics and dynamics and that for communications into local and global, and both of these communication contributions are again split into start-up time and time proportional to the length of the messages.

To produce the Figures, for each year the maximum equivalent number of grid points that could be processed within one hour on the then available computer was determined.

The next subsection shows the results obtained with the baseline configuration. After that, the results of several variations will be presented.

7.2   The baseline configuration

Panel (a1) of Figure (5) shows the maximum equivalent number of grid points for the current configuration of the ECMWF model (sisLsp) and an exEugp model. We address the sisLsp model first.

In panel (a1) of Figure (5) we note a sharp decrease after a maximum of 27220 grid points in 2027. The
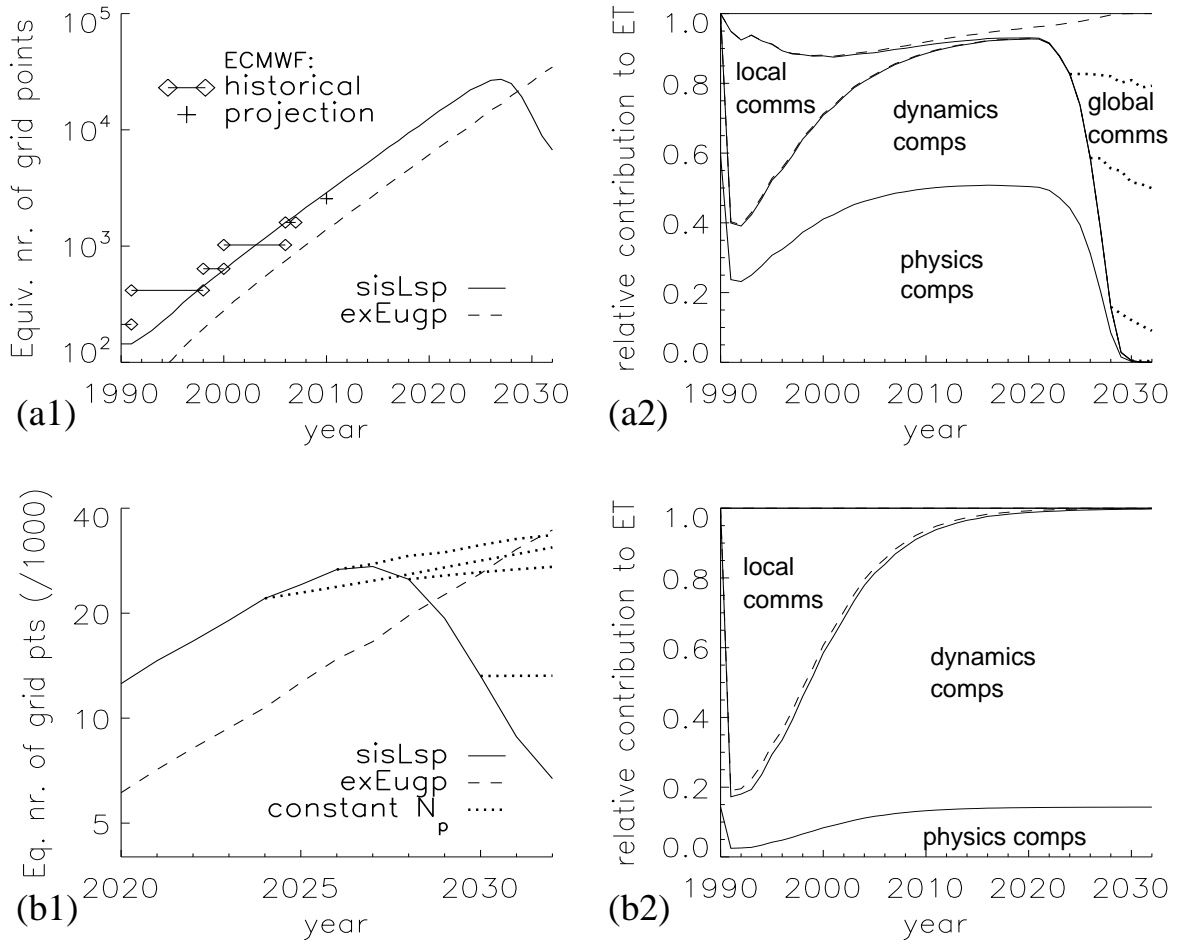
Figure 5: Results for the baseline configuration. Left panels: Achievable maximum equivalent number of grid points for the sisLsp and exEugp models; the lower panel (b1) is an enlargement of the period 2020 to 2032. Right panels: distribution of elapsed time over computations (physics and dynamics) and communications (local and global); the dashed lines split communication times into start-up time (lower area) and bandwidth-related (upper area); top panel for the sisLsp, bottom panel for the exEugp model. The dotted lines in panels (b1) and (a2) show the development in case the number of processors is kept constant, at the level of 2024, 2026, 2028 and 2030, resp., for the sisLsp model; in panel (b1) they show the development of the dynamics line, hence the separation of computation and communication. In panel (a1) the historical ECMWF *models and an* ECMWF *projection (T1279 in 2010) are also shown; the* ECMWF *model in 2006 was used to calibrate the sisLsp line.*

ECMWFmodel, being global, then has a resolution of 1.5 km. Panel (a2) of Figure (5) explains this decrease: the start-up time of global communications completely dominates the total elapsed time. The reason for this is three-fold: first, the horizontal extent of the computer becomes large, resulting in large delays for global communications due to the finite speed of light; second, because there are many processors, the number of messages has become large; third, computations have become very fast. The speed of computations has increased for two reasons: processors have become fast - so computations per grid point are fast; and they have become many - so the number of grid points per processor is small.

In reality, one would never buy a new computer with more and faster processors if its performance would be lower than the already available older computer. Hence, it is more realistic to assume that by the time that the

performance of the computer goes down, one would keep the number of processors fixed. In panels (b1) and (a2) of Figure (5) we have shown the result of this fixing the number of processors by the dotted lines. Note that we did allow the processors to become faster at the usual rate; so the only thing we changed was indeed the quantity of the processors, not their quality. As expected, in panel (b1) the decrease of number of grid points is now converted to a slow increase, in particular if the number of processors is kept constant at an early level (say of 2024). Panel (a2) shows that in that case the communication time remains limited as well, even though the improved compute performance still results in a slight shift from computation to communication.

In panel (a1) of Figure (5) we added the historical development of the ECMWF model from 1990 to 2006. The situation of 2006 was used for calibration, so it is no coincidence that in 2006 the sisLsp model is exactly on the full line. In the years before 2006, we see that the ECMWF model was above the line. In other words, the number of grid points in the ECMWF model increased at a lower pace than compute power. A possible explanation is that over the past 15 years, ECMWF has dedicated an increasing proportion of their compute resources to ensemble forecasting, thus leaving relatively fewer resources for their operational model. Another explanation is found in the observation that our performance model shows a large local communication cost in the early nineties (see panel (b1)). However, ECMWF employed shared memory machines in those times. So at ECMWF, less time was needed for communication than our performance model seems to suggest. Hence, more time was available for computation.

We tried to obtain some corroboration of our future estimates by a comparison to ECMWF's own projections. Because this information may influence their computer procurement, ECMWF is reluctant to provide this information directly, so we tried to make our own inference: ECMWF are currently making plans for their compute resources 2009–2012 [15]. Salmond [35] shows results for a T1279 computation, but without indicating when ECMWF would switch to that resolution. We combined these two numbers to assume that the ECMWF model could have a T1279 resolution in 2010. We marked this in panel (a1). This mark is close to but below the full line. Given the way we inferred this information, we conclude that ECMWF are assuming a similar increase in compute resources and resolution as we did, although there is some indication that ECMWF are expecting their model to grow less fast than compute resources, like they experienced in the past.

In Figure (5) we also added the results for a model without global communications, for which we chose an explicit Eulerian grid point model (exEugp). The panels (b1) and (b2) in Figure (5) show that this model does not suffer from the dominance of communication costs towards the end of the investigation period. The reason of course is found in the absence of global communications.

Mainly because of the large number of time steps, imposed by the CFL criterion, the explicit model has a far worse resolution than the sisLsp model over much of the period. However, after 2028 the sisLsp model becomes slower than this explicit model, because of its global communications, unless we do not let the number of processors grow from some earlier point in time. For example, the dotted line starting in 2026 and the dashed lines cross each other in 2032. in other words, in 2032 the exEugpmodel will be as efficient as the sisLsp model if the latter is run on fewer processors, namely the number that was available in 2026. Because the number of processors is assumed to grow with 50% per year (see Equation (17)), the number of processors needed for the sisLsp model is only 9% of that needed for the explicit model. So in 2032, probably the sisLsp model configuration is still preferable over an explicit model. But we observe that in order to run this model, the computer will then not be very advanced anymore; its number of processors is 9% of what a state-of-the-art computer in the field would have had.

We finish this subsection on the baseline experiment with a few remarks on Figure (5).

In 1990, our model for the computer configuration resulted in a system with 1 processor; hence, in 1990 there is no communication cost.

In the explicit model, the ratio of physics to dynamics computation is constant over time, because both their time steps are proportional to the resolution. For the sisLsp model, Figure (5) displays an almost constant ratio between physics and dynamics. In the course of time, dynamics will become slightly more expensive, because the Legendre transforms cost more than proportionality to the resolution would suggest. However, the Figure

shows that the effect is essentially negligible, which is a consequence of the slow logarithmic growth of this cost.

7.3   Some sensitivity experiments

Our computer configuration model contains many assumptions. We expect that even if they are valid nowadays, in 24 years the situation will have changed beyond their validity. Hence, it is relevant to check how our results depend on them.
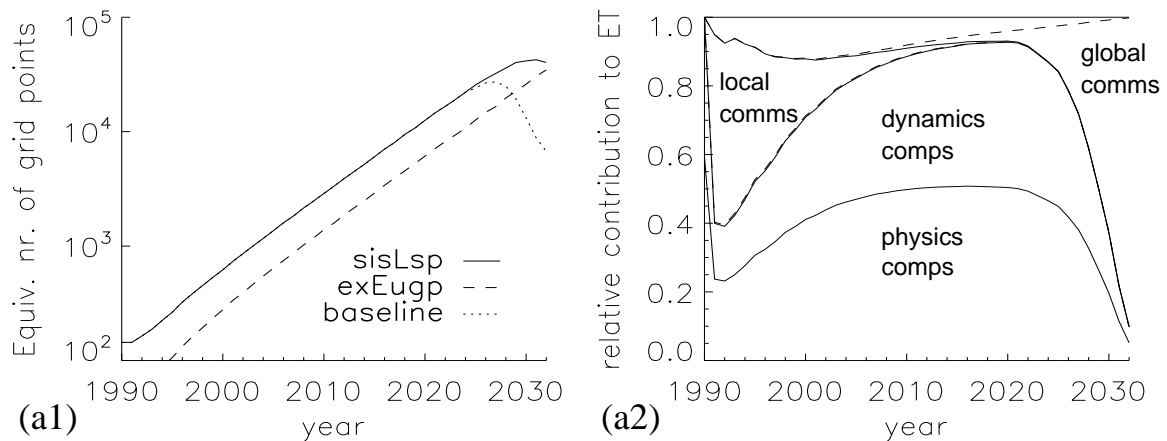

Figure 6: As Figure (5), panels (a1) and (a2), but for a computer with a footprint of 15 m. For reference, panel (a1) also reproduces the baseline, panel (a1) of Figure (5).

7.3.1   Footprint of the machine

Given the strong dominance of the start-up time of global messages after 2026, which is due to the finite speed of light in combination with the size of the footprint of the machine, the most relevant sensitivity is expected to lie in the model for the footprint size. In the baseline situation, it was taken dependent on the number of processors, with the distance between two processors maximised to 100 m. Figure (6) shows what happens if we fix that distance at 15 m, throughout the whole period. This distance gives the machine a floor space of approximately 2500 sq ft, which is the current configuration at Lawrence Livermore National Laboratory [27]. This implies a larger footprint than in the baseline before 2022, and a smaller one after 2022. The maximum resolution of the sisLsp model is now reached in 2031, at which time it has 21280 grid points in the x-direction. This is considerably more than in the baseline, where the maximum, 27220 grid points, will be reached in 2027. But clearly, the finite speed of light will become the determining factor. Albeit shifted by a few years, the overall picture did not change much. So our results are not very sensitive to the assumed size of the computer, in the sense that it may shift our conclusions by a few years.

7.3.2   Contribution of start-up to total communication time

Again observing the large impact of start-up time towards the end of the investigation period, we investigated the impact of the fairly arbitrary choice to distribute the communication costs as reported by Salmond [35] into 5% for start-up and 95% for (inverse) bandwidth. To this end, we reduced the start-up time to 1% of total communication costs. Figure (7) shows that the impact is negligible. The largest impact is seen in 2019,
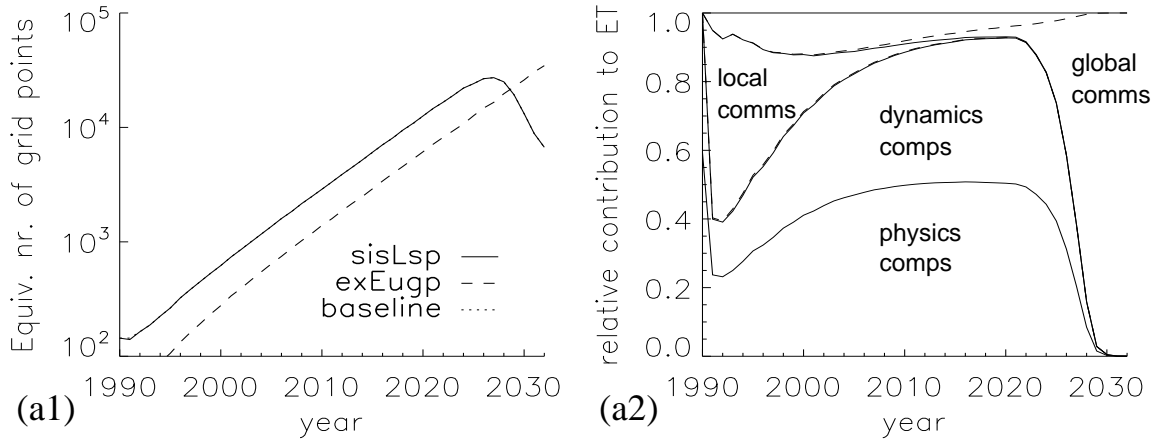
Figure 7: As Figure (5), top panels, but assuming latency accounts for 1% of communication time in 2006. The difference of sisLsp with the baseline cannot be discerned.

when the number of grid points increases by 84, to 10836. The reason for this negligible impact is of course the exponential growth of most parameters; whatever the initial conditions, the results are overwhelmingly determined by that growth.
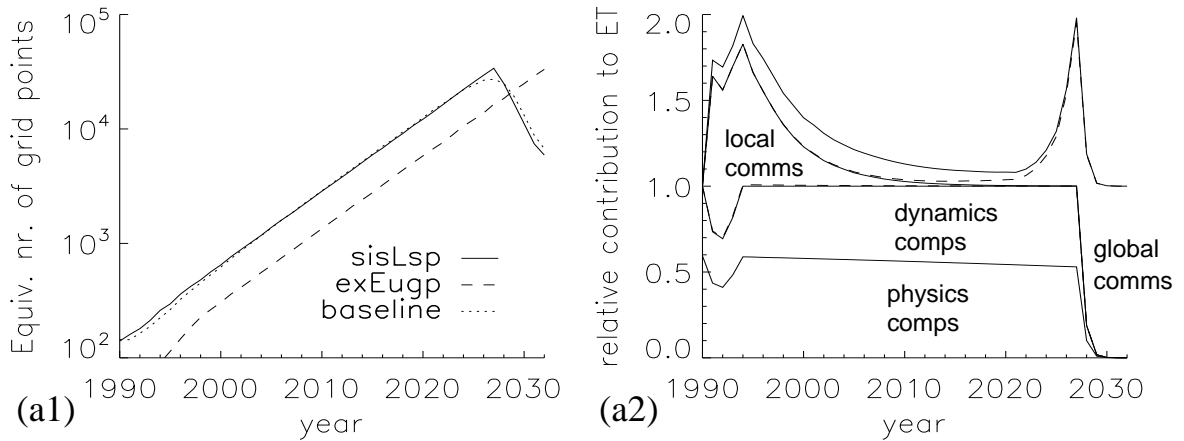
7.3.3   Communications overlap with computations

Figure 8: As Figure (5), top panels, but assuming communications overlap fully with communications.

In the baseline configuration, we assumed that communications and computations do not overlap at all. Figure (8) shows the results of our computer configuration model if they overlap completely. Because of the overlap, the relative contributions sum up to a number larger than 1. The maximum resolution, 33884 grid points, is reached in 2027. Apart from the sharper peak around 2027, the picture is very much the same as the baseline configuration.

7.3.4 Smaller growth of compute power

In the baseline configuration, we assumed that compute power grows at a rate of 20% per year, and the number of processors even at 50% per year. Most likely, power consumption will not allow this growth over the next 24 years. Hence, the next sensitivity test we put the growth of compute power to 0, but allow the number of processors still to grow at 50% per year. Figure (9) shows the results. The model performance lines are much flatter than in the baseline, but the overall appearance of the picture remains the same. The maximum resolution is 12860 grid points, reached in 2028.
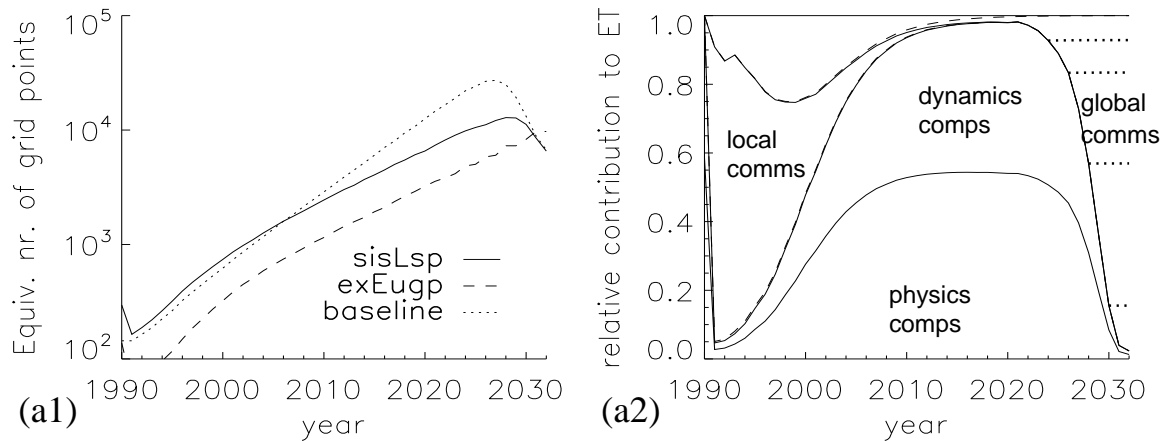

Figure 9: As Figure (5), top panels, but assuming no performance increase of single processors.

7.4 On a Grid computer

The current tendency is to try and run large applications on computers consisting of an aggregate of many relatively small machines [32], and in particular on Grid computers [18].

We adjusted our model to describe a Grid computer by the following modifications:

- Somewhere in the computer, two processors holding nearest neighbour data are separated by a distance of 30 km.

- The maximum distance for global communications is 3000 km.

- Start-up times of local and global communications are the sum, rather than the maximum of the time delay due to machine properties (Table (6.1)) and the delay due to the finiteness of the speed of light.

The last of these modifications was motivated by the observation that in our runs so far, the influence of the speed of light was totally negligible in the early years, but grew to total dominance within a year or two around 2027. By that time, the machine properties had become totally negligible due to the exponential decay with the constants in Table (6.1). However, in a Grid computer we expect that the effect will be important also in the earlier years, when the machine properties are not favourable yet.

Figure (10) shows the results of our runs for a Grid computer. The most obvious thing is that a Grid computer is totally unsuitable for a model with global communications, like the sisLsp model. After 2010, the communication costs in the sisLsp model have grown so big, and the number and speed of the processors so high, that each processor processes just one grid point. With higher number of processors, our parallelisation paradigm of
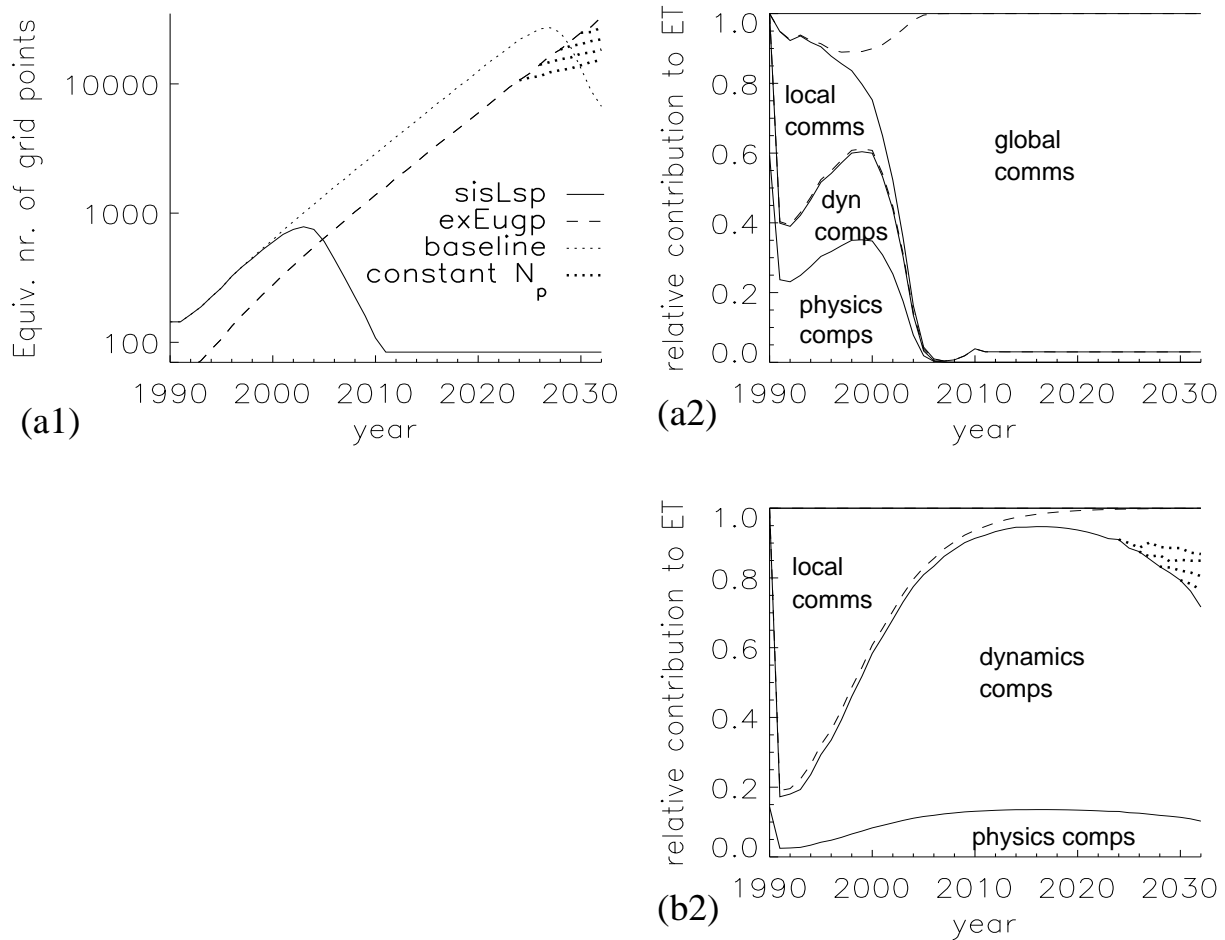
Figure 10: As Figure (5), but for a Grid computer. The dotted lines now show the situation for the explicit model if the number of processors is kept constant from 2024, 2026, 2028 and 2030 onwards, resp.

horizontal domain decomposition breaks down. That is why our model keeps the number of grid points in the sisLsp model constant after 2011; that number is 84 in the x-direction. The physical size of the Grid computer is then 580 km.

An explicit model, on the other hand, does hardly suffer from the larger distances between the processors, although in panel (b2) of Figure (10) we see that the speed of light becomes increasingly important towards the end of the investigation period. The increase of the contribution of the start-up time for local communications is much less abrupt than what we observed for global communications in the other Figures. This follows from the fact that the relative increase is solely due to the increase of the computational speed, hence decrease of computation time. Global communications, on the other hand, in addition suffer from increased communication distances and increased number of communications (factor $p_x - 1$ in Equation (11)).

The dotted lines in Figure (10), panel (b2) show that the communication times can be limited by keeping the number of processors constant from a certain date onwards, but the corresponding dotted lines in panel (a1) tell us that this strategy would result in a severe reduction of the number of grid points.

8   Discussion

Probably the most striking feature in the figures in the preceding section is the almost exponential growth of the

number of grid points, over most of the next 24 years, both for current state-of-the art models like sisLsp and models without any global communication (exEugp). Towards the end of the period, however, the finite speed of light reduces global communication speed so severely, that model performance deteriorates even though computers still grow faster and bigger.

We have shown that the overall behaviour of the maximum model resolution that can be achieved according to our model does not strongly depend on the choices of the many parameters in the model. ECMWF have made the right choices in their preference for a semi-implicit semi-Lagrangian spectral model, and in their abstinence from Grid computing.

In our model, we allow the number of processors and the speed of the processors to grow exponentially. According to this model, in 2032 a weather computer will have 3.9 million processors, packed in an installation covering $100 \times 100 \ m^2$. The speed of each processor will have increased 100 fold over 2006 values, *i.e.,* in the order of TFlops. We have not considered practical limits to number of processors, packing and housing, or processors speed, that may, *e.g.,* be imposed by constraints on power consumption. The only physical limit that we introduced in our model is the speed of light, limiting communication speed. In this, we took the optimistic view that communications would be at the speed of light, even though current technology of glass fibres and copper wires does not exceed 70% of that speed [23]. However, as with other parameters, our results do not substantially change with this reduced speed of light.

It is unlikely that a future computer system will have 4 million processors of the currently common design, if alone because it would require several power plants to provide the power to run such a machine. Power consumption constrains current processor design. Improved performance is now sought in multi-core processing. The question thus arises how our results would change if multi-cores are introduced. The first remark to be made is that in order to resolve the power problem, we should be talking not of dual cores or quads, but really of processors holding tens of thousands of cores. In principle, such multi-core processors fit seamlessly in our model of increased compute performance over time. It is not really relevant whether that performance increase is achieved by improving single core performance, or by sharing tasks over multi-cores. On communication, the main difference between a dual core processor and two single core processors is that the communication between the cores on a dual core processor is far less constrained by the speed of light. However, as soon as two or more multi-core processors are employed the local communication speed will be constrained by the biggest of the distances between any pair of neighbouring multi-cores. In that respect, there is no difference with a system solely built with single core processors. The situation of global communications is similar, although perhaps a machine with multi-core processors can be built with a much smaller footprint, thus enabling an extended use of *e.g.,* the sisLsp model design.

Other hardware developments are far more speculative, and hence it is difficult if not impossible to judge their impact on future computing. Amongst these, quantum computing may be the most challenging, although at this moment it is not clear at all whether a quantum computer can be used for an arbitrary algorithm [37]. Another challenging development is the "field programmable gate array", [16], which runs a program by "burning" it into hardware in stead of by executing assembly code. Numerical weather prediction is currently seen as one of the future core applications for such hardware [17]. Perhaps less far into the future is the use of graphics processors as power-efficient coprocessors [30].

Another issue that we did not attempt to address is the possible development of totally new algorithms for weather prediction. Until deep into the last century, the human forecaster beat numerical models. This is an indication that a system based on neural networks will not be able to beat NWP models, unless neural networks become much more intelligent than man. The lattice-gas method [10] will have a very hard task to describe fluid dynamics on the global scale. Interestingly, lattice-gas models may provide algorithms that can be implemented on quantum computers [43], and in the end we find that hard- and software develop jointly, in a direction that was not foreseen at all in this paper.

Even without straying that far from the currently common model structures, it is conceivable that a algorithm will be found that maintains the advantages of the semi-implicit method of large time steps while limiting

communications to local only. In particular, semi-implicit spectral element methods [9] may develop into that direction.

9   Conclusions

Based on the observed exponential growth of a range of parameters describing computer performance, both for computations and communications, we developed a model for the maximum resolution of two kinds of global numerical weather prediction models. The first kind is a semi-implicit, semi-Lagrangian, spectral model (sisLsp). This is currently the most commonly used model formulation. in particular, the ECMWF model uses it. The second kind is an explicit, Eulerian, grid point model (exEugp).

We did not constrain the exponential growth of the number of processors, and of the compute and communication performance parameters; but communication speed is constrained by the finite speed of light.

With our model, we have seen that the current sisLsp model formulation outperforms the explicit model by an order of magnitude on conventional computers, and will continue to do so for two decades to come, provided technological developments will indeed sustain exponential growth of single-site computers. An explicit model would over much of the period have a maximum resolution of approximately half that achieved with the sisLsp model. This corresponds to a time lag of 5 years. After about sixteen years from now, explicit models will be faster than models with global communication, if run on the same computer. After that, there will be a transitional period of several years, in which sisLsp models better be run on less advanced computers, and then even beat exEugp models on state-of-the-art computers. Eventually, on conventional computers, exEugp will outperform sisLsp, but this is not anticipated to happen within the investigated time frame of 24 years.

However, if technology develops more into the direction of Grid computing, an explicit model will soon outperform a model with global communications, provided that the physical distances between processors holding neighbouring data is kept small. This may require a careful data lay-out over the components of the Grid computer, in particular for a global model.

We found that the quality of the NWP models, as measured by the number of grid points that can be processed within a total elapsed time of 1 hour, will continue to grow, almost exponentially, until the communication delays become prohibitive. For a model with global communications this happens very suddenly, and we predict that to occur in 20 years time. A model without global communications is far less sensitive to communication delays, but, if implemented on a Grid computer, in 24 years time also such a model will start to suffer from the finiteness of the speed of light.

Although we did not explicitly model the possible development of multi-core processors, our conclusions do not depend on that development. But other hard- and software developments, like quantum computing and/or lattice gas methods, do not fit into our model, and hence may invalidate our conclusions. Given the speed of their developments, however, it is unlikely that they will mature enough within the next 24 years to become competitive with more conventional hard- and software.

10   Acknowledgements

A   Appendix: Some equations

In the main body of this paper we mention a number equations types that are encountered in NWP. In this Appendix, we write out several of them, for reference. For our purposes, the essential properties of the equations

are the communication requirements. Therefore, here it suffices to give the equations in their simplest forms; *e.g.,* the coordinate system will be local Cartesian, even though most NWP systems use more complicated systems. For example, usually, the horizontal coordinates are based on the spherical geometry, and the vertical coordinate often is normalised with surface pressure, to avoid that horizontal coordinate surfaces would be discontinuous in and across mountains.

A.1 The primitive equations

The formulation of the primitive equations in Cartesian coordinates is taken from [13]. We use the notation $\nabla_z = (\frac{\partial}{\partial x}, \frac{\partial}{\partial y})$, *i.e.,* the horizontal gradient operator. The horizontal velocity is denoted by $\boldsymbol{u}$; $\boldsymbol{k}$ is the vertical unity vector and f is the Coriolis parameter, describing the effect of rotation of the earth.

The first equation is the momentum equation:

$$\frac{d\boldsymbol{u}}{dt} + f\boldsymbol{k} \times \boldsymbol{u} + \frac{1}{\rho}\nabla_z p = S_{\boldsymbol{u}} \ . \tag{18}$$

In here, the time derivative is

$$\frac{d}{dt} = \frac{\partial}{\partial t} + \boldsymbol{u} \cdot \nabla_z + w\frac{\partial}{\partial z} \ . \tag{19}$$

The second equation expresses conservation of mass:

$$\frac{d\rho}{dt} + \nabla_z(\rho\boldsymbol{u}) + w\frac{\partial \rho w}{\partial z} = 0 \ . \tag{20}$$

The thermodynamic equation relates temperature change to change in pressure (usually connected to vertical velocity):

$$\frac{dT}{dt} - \frac{1}{\rho}\frac{dp}{dt} = S_T \ . \tag{21}$$

The equation of state for an ideal gas reads:

$$p = \rho R T \ . \tag{22}$$

In the hydrostatic approximation we have:

$$\frac{\partial p}{\partial z} = -\rho g \ . \tag{23}$$

In Equation (18) and Equation (21) we introduced source terms on the right hand side. Conventionally, the part of the NWP system that aims at estimating those terms is called the "physics", whereas the part that solves the equations without the source terms is called the "dynamics".

In the above equations, the horizontal derivatives require nearest neighbour communication in a grid point NWP model with horizontal domain decomposition.

In Equation (18) and Equation (20) the terms containing the partial time derivatives ($\frac{\partial}{\partial t}$) and the horizontal gradients ($\nabla_z p$, $\nabla_z \boldsymbol{u}$) sustain gravity waves, *i.e.,* the kind of wind waves we observe over water. Their phase speed is $c_g = \sqrt{gh}$, where h is a representative depth of the atmosphere (10 km); hence c_g is in the order of 300 m/s. The CFL criterion for numerical stability poses a severe restriction on the time step in an explicit time integration scheme if the wave speed is that high. By treating the mentioned terms with an implicit time-integration scheme the gravity waves can be slowed down considerably, but this will require the solution of a set of Helmholtz equations [13].

If the material time derivate ($\frac{d}{dt}$) is used in the form of Equation (19), the integration method is called Eulerian. If we then neglect all terms in the equations except those containing $\frac{d}{dt}$, we obtain a system that maintains advective waves. Their phase speed is the wind speed. The CFL criterion will limit the time step in relation to the maximum wind speed in the model domain, usually in the order of 100 m/s. Advective waves may be avoided by following an air parcel in its movement. In the equations, this is reflected by not using Equation

(19) to expand the material time derivative. Such integration schemes are called (semi-)Lagrangian. They allow much larger time steps. Because air parcels have to be traced upstream with the wind speed, and the time step may be big, advection may be from grid points several grid distances away. For communication this may imply that communication is needed not only with the nearest neighbours, but also with their neighbours and possibly with even further away processors. The communication is local, but over larger distances than in a Eulerian scheme.

A.2   The Helmholtz equation

The two-dimensional Helmholtz equation reads in Cartesian coordinates:

$$(\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} + m)\psi = \chi \ . \tag{24}$$

In here, ψ is the unknown, and χ is given. If the NWP model has N_x by N_y grid points, ψ and χ are vectors of length $N_x \times N_y$. The $N_x \times N_y$ by $N_x \times N_y$ matrix m is positive definite.

In a grid point NWP model with horizontal domain decomposition the numerical solution of a Helmholtz equation requires global communication. Solution methods may be direct inversion of a band matrix or indirect methods like conjugate gradients or multi-grid. Because spectral components are eigenvectors of the Helmholtz operator the solution of a Helmholtz equation in a spectral model is fast, and does not require any communication. But of course, in a spectral model global communications are needed to convert between the physical and the spectral grids.

References

[1] P. Bénard, J. Masek, and J. Vivoda, 2005: Stability of leap-frog constant-coefficients semi-implicit schemes for the fully elastic system of Euler equations. Case with orography. Mon. Wea. Rev, 133, 1065–1075.

[2] V. Bjerknes, 1904: Das Problem der Wettervorhersage, betrachtet vom Standpunkt der Mechanik und der Physik, Meteorol. Zeitschrift 21, 1–7.

[3] J. Boerhout, 2003: Reference HIRLAM scalability optimisation proposal, HIRLAM Newsletter 44, pp 22, http://hirlam.org/open/publications/NewsLetters/44/HIRLAMOptNewsletter.pdf.

[4] N. Brummer, V. Scarani, M. Wegmüller, M. Legré and N. Gisin, 2004: Direct Measurement of Superluminal Group Velocity and Signal Velocity in an Optical Fiber, Phys. Rev. Lett. 93, 203902.

[5] G.J. Cats and A.A.M. Wolters, 1996: The Hirlam Project, IEEE Computational Science and Engineering, Volume 3 , Issue 4, pp 4–7.

[6] J.G. Charney, 1951: Dynamical forecasting by numerical process, in Compendium of Meteorology, American Meteorological Society, Boston, 470âĂŞ-82.

[7] J. Côté and A. Staniforth, 1990: An accurate and efficient finite-element global model of the shallow water equations, Mon. Wea. Rev., 118, 2707–2717.

[8] D. Culler and J. Pal Singh, 1998: Parallel Computer Architecture: A Hardware-Software Approach. With Anoop Gupta. Morgan Kaufmann Publishers, 1998.

[9] J. Dennis , A. Fournier , W.F. Spotz , A. St-Cyr , M.A. Taylor , S.J. Thomas , H. Tufo, 2005: High-Resolution Mesh Convergence Properties and Parallel Efficiency of a Spectral Element Atmospheric Dynamical Core, International Journal of High Performance Computing Applications, 19, p.225–235.

[10] G.D. Doolen, 1991: Lattice Gas Methods: Theory, Applications, and Hardware. MIT Press.

[11] J. Dudhia, 1993: A nonhydrostatic version of the Penn State-NCAR mesoscale model: Validation tests and simulation of an Atlantic cyclone and cold front. Mon. Wea. Rev., 121, 1493–1513.

[12] The Dongarra Linpack benchmarks, and top 500 lists: `http:// www.top500.org`

[13] D.R. Durran, 1999: Numerical methods for wave equations in geophysical fluid dynamics, Springer Verlag.

[14] ECMWF, 2006: Supercomputer History, in `http://www.ecmwf.int/services/computing/overview/supercomputer_history.html`.

[15] ECMWF discussion papers, 2007: Presentations by the heads of Research and Operations to the Technical Advisory Committee.

[16] R. Enzler and M. Platzner, 2001: Application-driven design of dynamically reconfigurable processors Elektronische Daten (2001), available from Swiss Federal Institute of Technology (ETH), and from `http://e-collection.ethbib.ethz.ch/show?type=bericht&nr=369`.

[17] M. Feldman, 2008: A Modest Proposal for Petascale Computing, Editor's comment in HPCwire, 8 February 2008, `http://www.hpcwire.com/hpc/2112632.html`.

[18] I. Foster and C. Kesselman (Editors), 2004: The Grid 2: Blueprint for a New Computing Infrastructure, Elsevier.

[19] A.E. Gill, 1982: Atmosphere-Ocean Dynamics. Orlando, Academic Press.

[20] A.K. Gupta and W.J. Dally, 2006: Topology Optimization of Interconnection Networks, Computer Architecture Letters 5, 10–13.

[21] N. Gustafsson and A. McDonald, 1996: A comparison of the HIRLAM gridpoint and spectral semi-Lagrangian models, Monthly Weather Review 124, 2008–2022.

[22] G. J. Haltiner and F. L. Martin, 1957: Dynamical and physical meteorology. McGraw-Hill.

[23] J. Hecht, 2004: Understanding Fiber Optics, 4th ed., Prentice-Hall, Upper Saddle River, NJ, USA.

[24] J. Hennessy and D. Patterson, 2007: Computer Organization and Design, Elsevier.

[25] IBM Blue Gene specification sheet, `http://www-03.ibm.com/servers/deepcomputing/pdf/bluegene_spec_sheet.pdf`

[26] M. Jarraud, and C. Girard, 1984: An extensive quasi-operational comparison between a spectral and a grid-point model, ECMWF Seminar on Numerical Methods for Weather Prediction, 5–9 September 1983, Reading, England

[27] Lawrence Livermore National Laboratory Blue Gene configuration, `http://www.llnl.gov/asc/computing_resources/bluegenel/ configuration.html`

[28] A. McDonald and J.E. Haugen, 1993: A Two Time-Level, Three-Dimensional, Semi-Lagrangian, Semi-implicit, Limited-Area Gridpoint Model of the Primitive Equations. Part II: Extension to Hybrid Vertical Coordinates, Monthly Weather Review , 121, pp. 2077–2087.

[29] A. Noullez and M. Vergassola, A fast Legendre transform algorithm and applications to the adhesion model, Journal of Scientific Computing 9, 259–281.

[30] J. D. Owens, D. Luebke, N. Govindaraju, M. Harris, J. Krüger, A. E. Lefohn, and T. Purcell, 2007: A Survey of General-Purpose Computation on Graphics Hardware, Computer Graphics Forum, volume 26, pp. 80–113

[31] L.F. Richardson, 1922: Weather prediction by numerical process, Cambridge University Press.

[32] M. Ripeanu, 2006: Note on Zipf Distribution in Top500 Supercomputers List , IEEE Distributed Systems Online, (accepted), October 2006.

[33] H. Ritchie, C. Temperton, A. Simmons, M. Hortal, T. Davies, D. Dent, and M. Hamrud, 1995: Implementation of the Semi-Lagrangian Method in a High-Resolution Version of the ECMWF Forecast Model, Monthly Weather Review 123, pp. 489–514.

[34] D. Salmond, 2004: Computer Architectures and Aspects of NWP models, in: Proceedings of the ECMWF Seminar on Recent developments in numerical methods for atmospheric and ocean modelling, 6–10 September 2004, Reading, UK., pp 187–202.

[35] D. Salmond, 2007: Computational efficiency of the ECMWF forecasting system, in: Use of high performance computing in meteorology, Proceedings of the twelfth ECMWF workshop, 30 October - 3 November 2006, Reading. World Scientific, pp 1–12.

[36] D. Salmond, 2007: Private communication.

[37] P.W. Shor, 1994: Algorithms for quantum computation: Discrete logarithms and factoring, in Proceedings of the 35th Annual Symposium on Foundations of Computer Science, IEEE Computer Society Press (1994).

[38] W. C. Skamarock, J. B. Klemp, 2007: A Time-Split Nonhydrostatic Atmospheric Model for Research and NWP Applications, accepted for publication in J. Comp. Phys. special issue on environmental modeling.

[39] J. Steppeler, G. Doms, U. Schättler, H. W. Bitzer, A. Gassmann, U. Damrath and G. Gregoric, 2003: Meso-gamma scale forecasts using the nonhydrostatic model LM, Meteorology and Atmospheric Physics 82, 75–96.

[40] A.A.M. Wolters and G.J. Cats, 1993: A Parallel Implementation of the HIRLAM Model, in G.-R. Hoffmann and T. Kauranne (eds.), Parallel Supercomputing in Atmospheric Science, proceedings of the Fifth ECMWF Workshop on the Use of Parallel Processors in Meteorology, World Scientific Publ., 486–499

[41] A.A.M. Wolters, R. A. van Engelen, G.J. Cats, N. Gustafsson, and T. Wilhelmsson, 1994: A Data-Parallel HIRLAM Forecast Model, in proceedings of the Sixth ECMWF Workshop on the Use of Parallel Processors in Meteorology, Nov 21–25, Reading, UK, World Scientific, Singapore etc., 49–62

[42] A.A.M. Wolters, G.J. Cats, N. Gustafsson, and T. Wilhelmsson, 1995: Dataparallel Semi-Lagrangian Numerical Weather Forecasting, in proceedings of Frontiers '95, the Fifth Symposium on the Frontiers of Massively Parallel Computation, February 6–9, McLean, Virginia, USA, IEEE Computer Society Press, 164–170

[43] J. Yepez, 2001: Quantum lattice-gas model for computational fluid dynamics, Phys. Rev. E 63, 046702, pp 18.