Stability of Numerical Schemes for PDE's.

Rodolfo R. Rosales.*

MIT, Friday February 12, 1999.

Abstract

The purpose of these notes is to give some examples illustrating how naive numerical approximations to PDE's may not work at all as expected. In addition, the following two important notions are introduced: **(I) von Neumann stability analysis** — helps identify when (and if) numerical schemes behave properly. **(II) Artificial viscosity** — a tool in stabilizing numerical schemes. These notes should be read in conjunction with the use of the **MatLab scripts** (in the Athena 18311-Toolkit at MIT) whose names end with the acronym GBNS (for Good-Bad-Numerical-Schemes).

Contents

List of Figures

*MIT, Department of Mathematics, room 2-337, Cambridge, MA 02139.

1 Naive Scheme for the Wave Equation.

We will illustrate the points we want to make with the wave equation (in one space dimension)

$$\frac{\partial^2 u}{\partial t^2} - \frac{\partial^2 u}{\partial x^2} = 0\,. \tag{1.1}$$

Since this equation is second order in time, it needs two initial conditions. For example:

$$u(x,0) = u_0(x) \quad \text{and} \quad \frac{\partial u}{\partial t}(x,0) = v_0(x)\,. \tag{1.2}$$

We will assume here that both u_0 and v_0 are periodic, with some **period** $\boxed{T > 0.}$ Then the solution of (1.1) is periodic in x with the same period: $\boxed{u(x+T,t) = u(x,t).}$

Remark 1.1 *We note that, in fact, we can write the solution of this problem explicitly*

$$u = \frac{1}{2}\left(u_0(x-t) + u_0(x+t) + \int_{x-t}^{x+t} v_0(s)ds\right).$$

However, this is not the point here (see below).

Operate now as if (1.1) were complicated enough that we needed to solve the equation numerically. For this purpose introduce a **numerical grid** $\{x_n, t_j\}$ — where n and j are integers, as follows

$$x_n = x_0 + n\Delta x \quad \text{and} \quad t_j = j\Delta t\,. \tag{1.3}$$

Here Δx and Δt are some "small" positive constants and x_0 is arbitrary. Next replace the function $u = u(x,t)$ of the continuum variables x and t by a **discrete double sequence** $\{u_n^j\}$, where

$$u_n^j = u(x_n, t_j)\,. \tag{1.4}$$

Finally, introduce the new variable $v = \dfrac{\partial u}{\partial t}$ to re-write equation (1.1) as a first order in time system

$$\frac{\partial u}{\partial t} = v \quad \text{and} \quad \frac{\partial v}{\partial t} = \frac{\partial^2 u}{\partial x^2}\,. \tag{1.5}$$

In view of (1.4) it is now clear that u_n^j (and the similarly defined v_n^j) should satisfy

$$\frac{u_n^{j+1} - u_n^j}{\Delta t} = v_n^j + O(\Delta t) \quad \text{and} \quad \frac{v_n^{j+1} - v_n^j}{\Delta t} = \frac{u_{n+1}^j - 2u_n^j + u_{n-1}^j}{(\Delta x)^2} + O(\Delta t, (\Delta x)^2)\,, \tag{1.6}$$

which can be checked by expanding u_n^{j+1}, u_{n+1}^j, ... in Taylor series centered at (x_n, t_j) — using (1.4) — and substituting the expansions in (1.6). This suggests the following numerical scheme, allowing simple calculation of the solution at time $t = t_{j+1}$ (once it is known at time $t = t_j$)

$$u_n^{j+1} = u_n^j + \Delta t\, v_n^j \quad \text{and} \quad v_n^{j+1} = v_n^j + \frac{\Delta t}{(\Delta x)^2}\left(u_{n+1}^j - 2u_n^j + u_{n-1}^j\right), \tag{1.7}$$

where the errors should be of size $O(\Delta t, (\Delta x)^2)$, that is: small.

Upon implementation one quickly discovers that **this algorithm is disastrously bad.** The MatLab scripts: InitGBNS, lectureGBNS, demoGBNS, movieGBNS and the help file readmeGBNS in the Athena 18311-Toolkit all deal with this scheme and another one to be introduced later in these notes. In particular, lectureGBNS goes through and explains a series of calculations showing the details of how the scheme fails. We illustrate here the problem with a couple of examples.

Example 1.1 *Consider the following initial data (with period $T = 2$) for equation (1.5):*

$$u(x, 0) = u_0(x) = \frac{1}{2}\,(1 + \cos(\pi\, x)) \quad \textit{and} \quad v(x, 0) = v_0(x) \equiv 0\,. \tag{1.8}$$

The exact solution: $\boxed{u = \frac{1}{4}\,(2 + \cos(\pi\,(x - t)) + \cos(\pi(x + t))) = \frac{1}{2}\,(1 + \cos(\pi\, x)\,\cos(\pi\, t))}$ *— see remark 1.1 — is clearly also periodic in time of period 2 (a standing wave). For the numerical solution we take* $\boxed{\Delta x = 2\,\Delta t = 2/N}$ *(for some "large" N) and* $\boxed{x_0 = -1}$ *in (1.3). Then we implement (1.7) for $1 \le n \le N$ (the periodicity of the solution means that the indexes $n + N$ and n are equivalent) and solve the equations over one time period:* $\boxed{0 \le t \le 2.}$

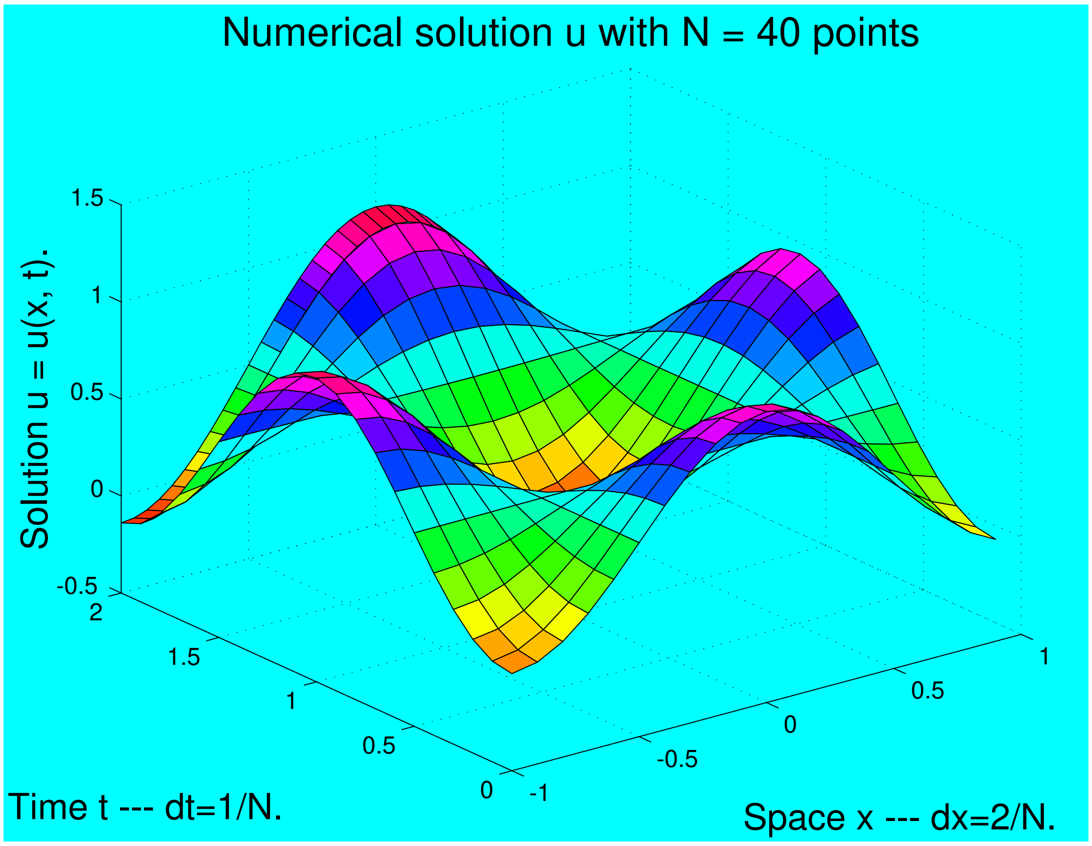

Figure 1.1: Solution of (1.5) with initial data (1.8) using (1.7) with 40 points in the space grid. To avoid an over-dense graph not all the points in the numerical grid are plotted. However, enough points to show all the relevant details are kept.

Figure 1.1 shows the result of this calculation using $N = 40$. Note that the periodicity in time fails to hold. In fact, after one time period the numerical method appears to have **amplified** *the initial* *data by about 30%! However, maybe this is not so bad (***or is it?***); after all the value of N being used is not that large and the numerical solution looks otherwise quite reasonable.*

Let us now check what happens as we increase the resolution (larger N). Any reasonable numerical scheme ought to give a better approximation when we do this. Figure 1.2 shows the result of increasing N to $N = 57$ (a rather small increase). The new approximation is not only not better; it is a **disaster***. By time $t \approx 2$, $O(1)$ grid scale (i.e. wavelength $= 2\,\Delta x$) oscillations appear in the numerical solution, making it useless. As we will soon see, the scheme is amplifying the errors; the 30% amplification of the initial cosine wave seen when using $N = 40$ was just a forewarning of what happens for larger N. As N is made even larger, the oscillations generated become huge (in fact,*

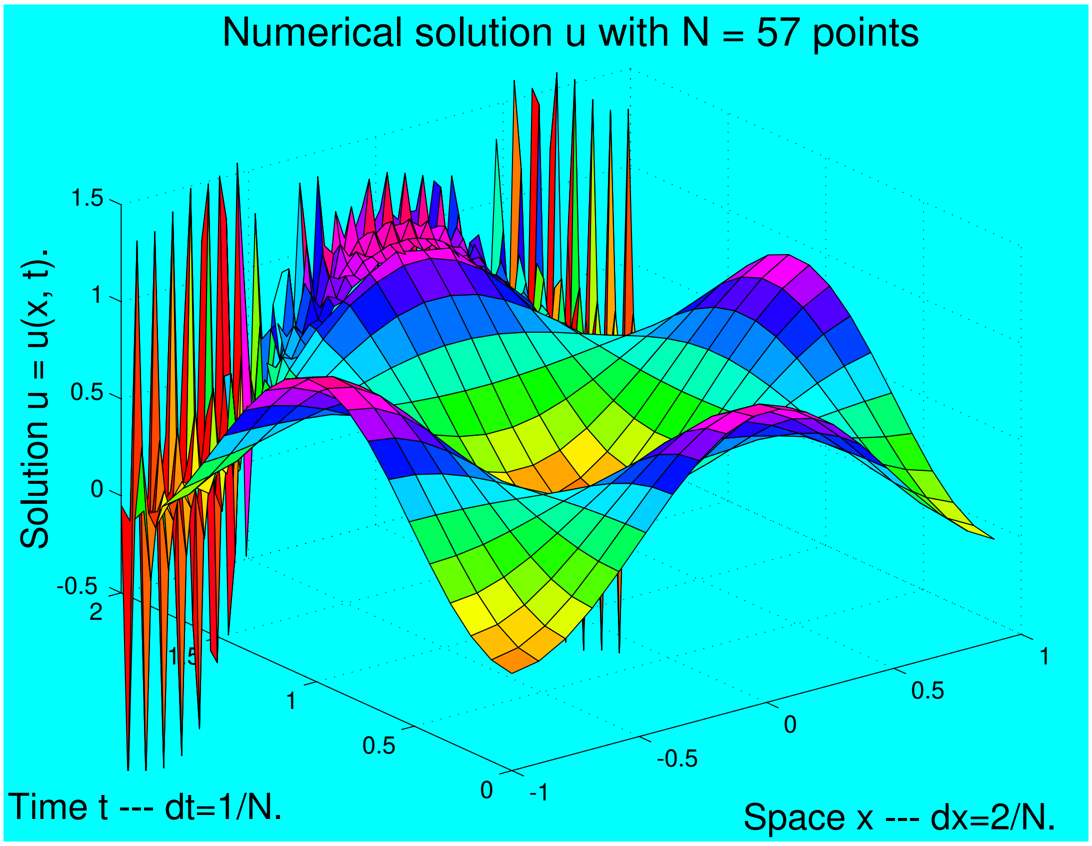

Figure 1.2: Solution of (1.5) with initial data (1.8) using (1.7) with 57 points in the space grid. To avoid an over-dense graph not all the points in the numerical grid are plotted. However, enough points to show all the relevant details are kept.

their size increases exponentially with N, as we will soon show). This is illustrated by figure 1.3, which corresponds to $N = 80$. Here (instead of a 3D graph) we plot the numerical solution at time $t = 2$. Grid scale (wavelength $= 2\Delta x$) oscillations is all that can be seen in this graph — notice the (very large) vertical scale on this figure!

Finally, we point out that if (instead of increasing N) we compute for longer times, the same effect of large amplitude grid scale oscillations arising (which grow exponentially in time) is observed.

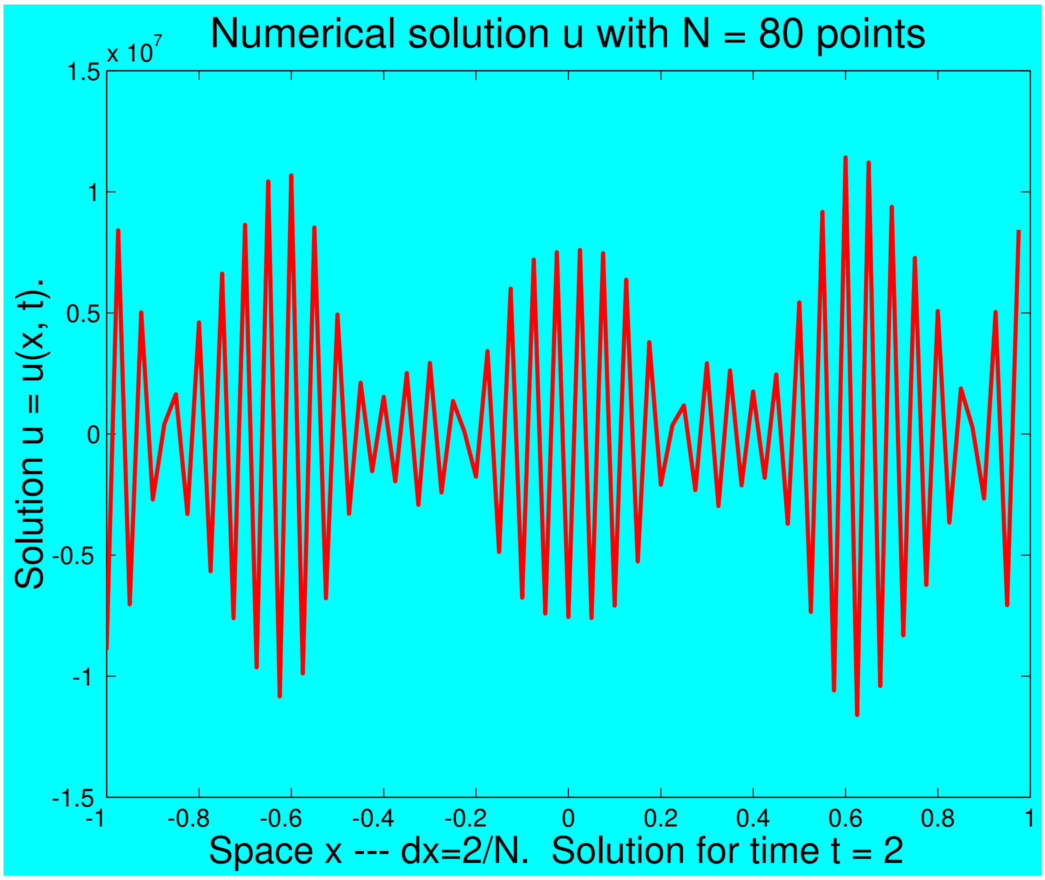

Figure 1.3: Solution of (1.5) with initial data (1.8) using (1.7) with 80 points in the space grid. Notice the large amplitude grid scale oscillations generated by the scheme. There is nothing but numerical noise in this picture!

Example 1.2 *In a second example we take the following Gaussian initial data for equation (1.5)*

$$u(x, 0) = u_0(x) = \exp(-a \ln(10)\, x^2) \quad \textit{and} \quad v(x, 0) = v_0(x) \equiv 0\,, \tag{1.9}$$

for $-1 \leq x \leq 1$, where $a > 0$ is a constant. We extend this to periodic initial data (of period $T = 2$) by repeating the above profiles over each interval $(2n-1) \leq x \leq (2n+1)$, with n integer. These initial values are not smooth — as were the ones in the prior example. There is a small corner in $u_0(x)$, whenever x is an odd integer (in particular for $x = \pm 1$). This is because at these points there is a cut-off from a Gaussian centered at $x - 1$ to one centered at $x + 1$. Notice that the size of the miss-match in the derivatives of u_0 goes down very rapidly as a increases.

For the numerical solution we take $x_0 = -1$, $\Delta x = 0.02$ *and* $\Delta t = 0.01$ *in (1.3) — this corresponds to* $N = 100$ *in the notation of example 1.1 — and use (1.7) to solve the equations for* $0 \le t \le 0.5$. *This is very similar to what we did in the prior example, except that here we vary the initial conditions (by changing the parameter a) instead of changing the resolution with variations in* N.

In the first calculation, we take a relatively large a, namely $a = 10$. *Figure 1.4 shows the result of this calculation, which appears quite reasonable.*

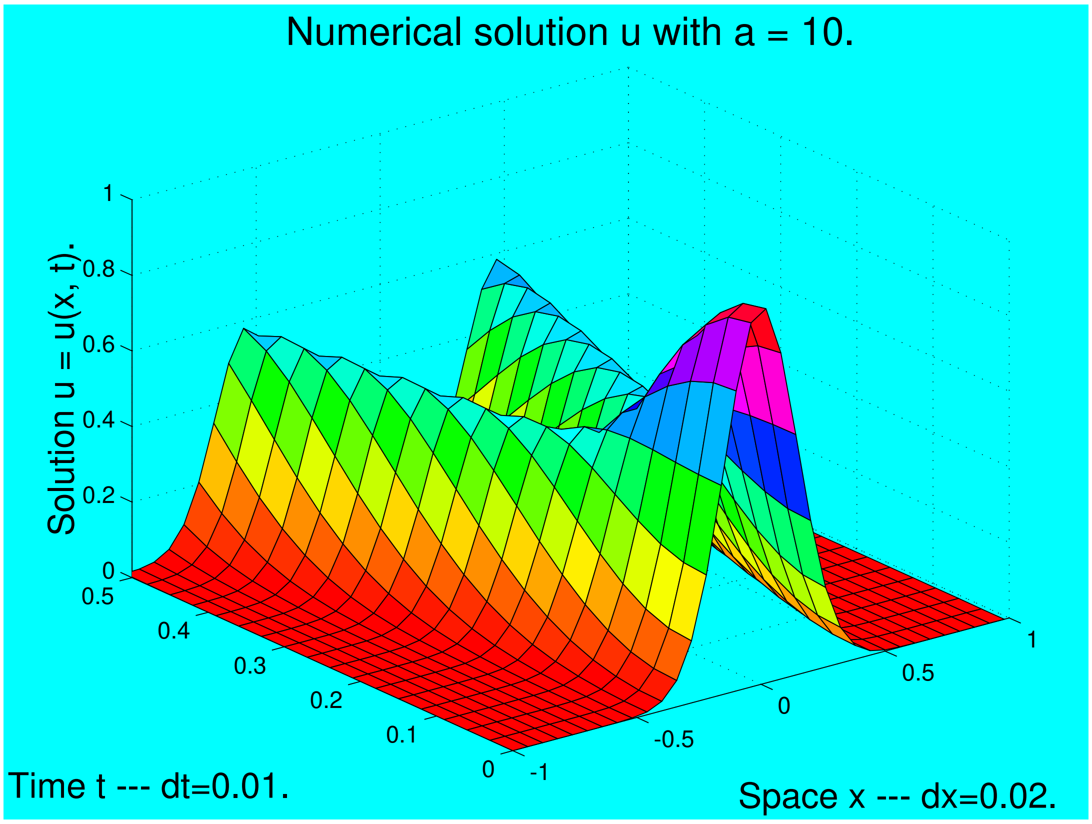

Figure 1.4: Solution of (1.5) with initial data (1.9) using (1.7) with 100 points in the space grid and $a = 10$. To avoid an over-dense graph not all the points in the numerical grid are plotted (enough points to show all the relevant details are kept).

In the second calculation, we take a smaller value $a = 6$. *This makes the corners more substantial (though still pretty weak). Figure 1.5 shows the result of this last calculation, which is now not reasonable at all. It is quite clear that, just as in the prior example, the small errors that are triggered by the corners are amplified by the scheme (so we observe grid scale oscillations near* $x = \pm 1$ *towards the end of the run).*

Finally, we point out that, if the calculations are run for times longer than $0 \le t \le 0.5$, *even the one with* $a = 10$ *eventually shows grid scale oscillations. These grow exponentially in time and pretty soon dominate the whole solution (not just the neighborhood of* $x = \pm 1$*) with huge amplitudes.*

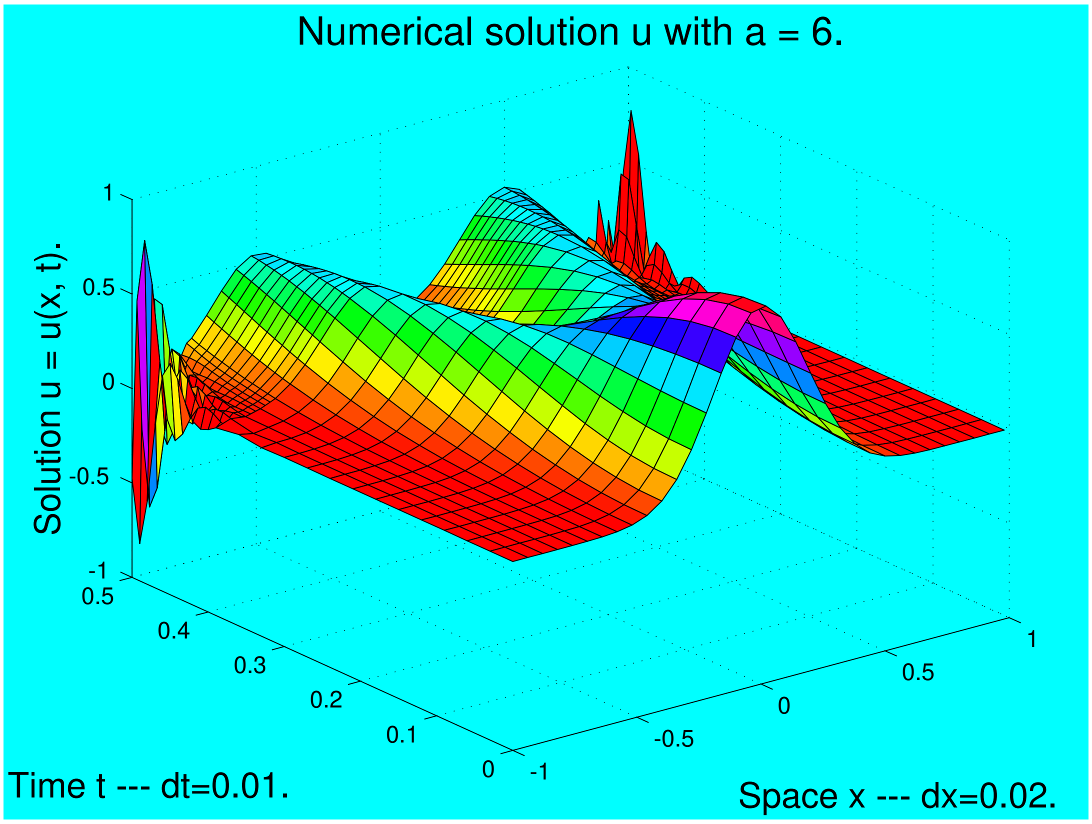

Figure 1.5: Solution of (1.5) with initial data (1.9) using (1.7) with 100 points in the space grid and $a = 6$. To avoid an over-dense graph not all the points in the numerical grid are plotted (enough points to show all the relevant details are kept).

The next section gives a detailed explanation of why this is happening.

2 von Neumann stability analysis for PDE's.

In this section we introduce the von Neumann stability analysis technique, that can be used to analyze numerical schemes and predict when the behavior observed in the prior section will occur. There are two **basic** concepts useful in understanding numerical schemes. These are the notions of **consistency** and **stability.** For a numerical scheme to be useful it must be both consistent and stable. It is very important to realize that these two notions are **independent**.

Consistency simply means that, as Δx and Δt vanish, the solutions of the equation must satisfy the numerical scheme with errors that vanish. This is in fact what equation (1.6) tells us about the scheme in (1.7). Consistency guarantees that the scheme truly approximates the equation we intend to solve with it (and not something else).

Stability simply means that the scheme does not amplify errors. Obviously this is very important, since errors are impossible to avoid in any numerical calculation. In fact, even in the ideal case of infinite precision, we still have to deal with discretization errors — i.e. the O terms in (1.6). Clearly, if errors are amplified, pretty soon they will dominate any computation (making it useless).

As it turns out, **for linear constant coefficient schemes such as (1.7), a complete stability analysis is possible,** because the numerical algorithm equations can be solved exactly by separation of variables. This means then that any solution of the scheme can be written as a superposition of Fourier modes. These Fourier modes are solutions of the form

$$u_n^j = U\, G^j\, e^{ikn} \quad \text{and} \quad v_n^j = V\, G^j\, e^{ikn}\,, \tag{2.1}$$

where U, V, G and k are constants (with k real). Generally double sequences like this will be solutions provided G, U and V are restricted by some functional relations of the form $G = G(k, \Delta x, \Delta t)$, $U = U(k, \Delta x, \Delta t)$ and $V = V(k, \Delta x, \Delta t)$ — below we carry through the calculations for the specific example of (1.7).

G is called the **Growth Factor**. It is clear that:

$$\boxed{\textbf{for stability } \|G\| \leq 1 \textbf{ is needed for all } k.} \tag{2.2}$$

Else some modes will be amplified by a factor G in each time step, eventually dominating the solution. A scheme is called **stable** if the stability condition $\|G\| \leq 1$ can be satisfied with (perhaps) a restriction on the time step of the form $0 < \Delta t \leq \tau(\Delta x)$, where τ is a **positive** function of its argument. Notice that restrictions of this latter form allow arbitrarily small time and space steps, which are needed to be able to compute the solution with any required degree of accuracy (how small is determined by how well consistency is satisfied, which determines the size of the errors for any given Δt and Δx).

Remark 2.1 *The parameter k is the* **wavenumber** *of the mode, related to the* **wavelength** *λ in space*[1] *by $\lambda = (2\pi\Delta x)/k$. For the particular case of* **periodic problems** *(such as the ones considered in examples 1.1 and 1.2), the Fourier modes (2.1) must also satisfy the periodicity condition. That is, one must have $\lambda = T/\ell$, where ℓ is an integer and T is the period in space. Since in this case one would normally take $\Delta x = T/N$, where N is a large natural number, the acceptable values for k end up restricted to the set*

$$k = k_\ell = \frac{2\,\pi\,\Delta x}{T}\,\ell = \frac{2\pi}{N}\,\ell \quad \textit{and} \quad \lambda = \lambda_\ell = \frac{T}{\ell}\,, \quad \textit{with} \quad 0 \leq \ell < N\,. \tag{2.3}$$

Here the upper bound N on ℓ follows from the fact that k_ℓ and $k_{\ell+N}$ give the same Fourier mode in (2.1); thus there is no reason to keep both.

We note that (due to the fact that the numerical scheme only samples the solution at a discrete set $\{x_n\}$ of points in space) there is a certain trickiness in the **interpretation of the wavelengths** *λ_ℓ above. Clearly, $\ell = 0$ corresponds to a solution independent of x and $\ell = 1$* corresponds to the fundamental mode with wavelength T in x. *As ℓ continues to increase harmonics of this fundamental mode appear, with wavelengths $T/2$, $T/3$ … However, this process cannot continue forever, since*

[1] Write the argument kn in the exponentials in (2.1) as $kn = \dfrac{k}{\Delta x}(x_n - x_0)$, using (1.3).

the numerical grid cannot resolve arbitrarily small wavelengths. In fact, the shortest wavelength that can be resolved corresponds to $\ell = N/2$ with $\lambda_\ell = 2\,\Delta x$ (grid size oscillations, with period 2 in n: the solution alternates between two values on the grid). *To see this recall that k_ℓ and $k_{\ell+N}$ give the same Fourier mode in (2.1). Thus the mode $(N-\ell)$ has the same wavelength as the mode $-\ell$, i.e. T/ℓ. This means that, after $\ell = N/2$ the wavelengths start increasing, to reach back the fundamental mode at $\ell = N-1$. Each wavelength then actually appears twice in the range $1 < \ell < N$.*

We should not be too surprised by the fact that each wavelength appears twice in the range $1 < \ell < N$. Notice that the modes in (2.1) are complex valued (except when k is a multiple of 2π). Thus, to be real valued any solution should include both the modes and their complex conjugates. However, the mode conjugate to the one with $k = k_\ell$ above in (2.3) is the mode with $k = k_{-\ell}$, which is precisely the same as the mode with $k = k_{N-\ell}$.

In any numerical calculation it is the modes with wavelengths of the order of the grid size Δx (i.e. ℓ close to $N/2$) that are worrisome in terms of instabilities. These modes cannot be expected to represent accurately any true feature of the real solution one is trying to compute[2] *and should not have any significant presence in the numerical solution. Thus, it is very important that they not be amplified by the scheme. In fact, generally it is desirable to have them damped, since they mostly represent numerical "noise" generated by all the approximations implicit in any numerical calculation.*

On the other hand, the modes with wavelengths much bigger than Δx (that is, $\ell \approx 0$ or $\ell \approx N$ in (2.3)) should be treated "accurately" by the scheme. By this we mean that their time evolution (given by the factors G^j in (2.1)) should be as close as possible to the one provided by the PDE the scheme approximates. This is what consistency is all about.

Consider now the special case of the algorithm (1.7). To see under which conditions (2.1) is a solution, substitute this form into (1.7). Dividing by the common factor $G^j\, e^{ikn}$ it follows that

$$G\,U = U + \Delta t\,V \quad \text{and} \quad G\,V = V + \frac{\Delta t}{(\Delta x)^2}(e^{ik} - 2 + e^{-ik})\,U\,.$$

Clearly an eigenvalue equation $\mathbf{A}\,\mathbf{Y} = G\,\mathbf{Y}$, with eigenvalue G, eigenvector $\mathbf{Y} = (U, V)^T$ and matrix of coefficients

$$\mathbf{A} = \begin{pmatrix} 1 & \Delta t \\ -4\frac{\Delta t}{(\Delta x)^2}\sin^2(\frac{k}{2}) & 1 \end{pmatrix}.$$

From the characteristic equation $\det(\mathbf{A} - G) = 0$, then

$$G = 1 \pm 2\,i\,\frac{\Delta t}{\Delta x}\sin(\frac{1}{2}k)\,. \tag{2.4}$$

It is clear that, **for (1.7) there is no stability,** since (2.4) yields

$$\|G\|^2 = 1 + \left(2\,\frac{\Delta t}{\Delta x}\sin(\frac{1}{2}k)\right)^2, \tag{2.5}$$

which is always bigger than one.

[2]Recall (1.4), which makes sense in terms of approximating the solution only if Δx is much smaller than any distance over which the solution changes significantly.

Notice that the **maximum amplification for the scheme (1.7) occurs** — as follows from (2.5) — **for $k = \pi$.** This corresponds to $\ell = N/2$ in (2.3), i.e.: **grid size oscillations with $\lambda = 2\,\Delta x$.** In this case

$$\|G\| = G_M = \sqrt{1 + 4\,\eta}\,, \tag{2.6}$$

where $\boxed{\eta = (\Delta t/\Delta x)^2\,.}$ For (1.7), the amplitude of the grid size oscillations grows like G_M^j. Thus we can write for the amplification factor $A_2 = A_2(t)$ (for the period $2\Delta x$ mode)

$$A_2 = \exp(t\frac{\ln(G_M)}{\Delta t})\,, \tag{2.7}$$

where we have used $j = t/\Delta t$. In particular (in **examples 1.1 and 1.2** earlier) we took $\Delta x = 2\,\Delta t$ and $\Delta t = 1/N$, so that

$$A_2 = \exp(\frac{\ln 2}{2}\,N\,t) = 2^{\frac{N\,t}{2}}\,. \tag{2.8}$$

We will now use these results to explain the behavior observed earlier in figures 1.1 through 1.5.

Remark 2.2 *Consider first* **example 1.1,** *with the initial data for scheme (1.7) given by*

$$u_n^0 = \frac{1}{2}\left(1 - \cos(\frac{2n\pi}{N})\right) \quad \textit{and} \quad v_n^0 = 0\,.$$

These data correspond to a superposition of just three modes in (2.1), with $k = k_0$, $k = k_1$ and $k = k_{-1} \sim k_{N-1}$ in (2.3). Thus, the **exact solution for the scheme equations** *is rather simple and has the form*

$$u_n^j = \frac{1}{2}\left(1 - \frac{g^j + \bar{g}^j}{2}\cos(\frac{2n\pi}{N})\right) \quad \textit{and} \quad v_n^j = \frac{g^j - \bar{g}^j}{2i}\,\hat{v}\,\cos(\frac{2n\pi}{N})\,, \quad \textit{for} \quad g = 1 + i\sin(\frac{\pi}{N})\,, \tag{2.9}$$

where $\hat{v}$ is a constant and $\bar{g}$ denotes the complex conjugate of g. Of course, g and $\bar{g}$ are the values G in (2.4) takes for $k = k_1 = 2\pi/N$.

Notice that the exact solution (2.9) does not exhibit any catastrophic growth of grid size oscillations, as was observed in example 1.1. However, the results displayed in figures 1.1 through 1.3 do not correspond to the exact solution above but to actual computations using the scheme in (1.7) — which were done using double precision floating point arithmetic (MatLab's default). The round off errors introduced by the finite precision of the calculations introduces (very small) perturbations into the exact solution above, which the scheme then evolves in time just as if they were part of the solution.

To understand what the scheme does with the perturbations introduced by the finite precision, decompose them into a sum over the modes in (2.1). This sum will generally include **all** *the modes, in particular the highly amplified ones with grid size wavelengths. Consider then what would happen with the solution of the scheme if we add to the initial data above*[3] *a small amount of the component corresponding to the maximum amplification rate above in (2.6). Let the amplitude of this component be ϵ, where ϵ has (roughly) the size of the expected errors. Actually, ϵ should be a little smaller than the round off errors that occur, since not all the errors get projected into the fastest growing modes. Thus take* $\boxed{\epsilon = O(10^{-17})}$ *as a* **good ballpark figure for the calculations in section 1** *and use (2.8) above to explain the behavior observed in figures 1.1 through 1.3, as follows:*

[3] Which has only components corresponding to $\ell = 0$, $\ell = 1$ and $\ell = N - 1$ in (2.3).

1. *First, for $N = 40$, (2.8) gives $A_2 \approx 1.1 \times 10^{12}$ for the final time $t = 2$. This is not enough to compensate for the smallness of ϵ and the numerical solution is well described by (2.9).*

   Notice that (2.9) is not periodic in time; since the wave amplitude in u behaves like $Re(g^j)$, which grows as j grows. In fact, $2\,N = 80$ steps are needed to reach the final time $t = 2$ and it is easy to check that

$$Re(g^{80}) = Re\left\{\left(1 + i\,\sin(\frac{\pi}{40})\right)^{80}\right\} \approx 1.28\,.$$

   This agrees quite well with the $\approx 30\%$ growth in the wave amplitude observed in figure 1.1.

2. *Second, for $N = 57$, (2.8) gives $A_2 \approx 1.4 \times 10^{17}$ for the final time $t = 2$. This is about the same as ϵ^{-1} and agrees with the fact that grid oscillations of $O(1)$ amplitude are observed in figure 1.2.*

3. *Third, for $N = 80$, (2.8) gives $A_2 \approx 1.2 \times 10^{24}$ for the final time $t = 2$. This is about 10^7 times bigger than ϵ^{-1}, which (again) agrees pretty well with the observed amplitude of the grid size oscillations in figure 1.3.*

4. *Finally, it is not just the mode with $\ell = N/2$ in (2.3) that gets a large amplification factor by the scheme. All the ones with $\ell \approx N/2$ do and should thus be present in the solution. It is well known that when sinusoidals with close wavenumbers are added, "beats" with wavenumbers equal to the difference in wavenumbers occur. Thus, in this case we should observe "beats" with wavenumbers low multiples of $k_1 = 2\pi/N$ — which, indeed, are quite obvious in figure 1.3.*

Remark 2.3 *Now consider* **example 1.2,** *where $N = 100$ and $0 \le t \le 0.5$. Then, for the time $t = 0.5$, equation (2.8) gives* $\boxed{A_2 \approx 3.4 \times 10^7.}$

In this case the initial data has components in all the modes $0 \le \ell < N$ in (2.3). In fact, because of the corners at $x = \pm 1$, the amplitude present in the higher modes is relatively large. The strength of these corners can be measured by the jump in the derivative of the initial data *there:* $\boxed{J(a) = 4\,a\,\ln(10)\,10^{-a}.}$ *For moderate[4] size a, $J(a)$ pretty much determines how much amplitude there is in the higher modes. Now $J(10) \approx 9.2 \times 10^{-9}$ and $J(6) \approx 5.5 \times 10^{-5}$. Thus, from the value of A_2 above, it should be clear why in figure 1.4 (corresponding to $a = 10$) the solution exhibits no detectable oscillations, while in figure 1.5 (corresponding to $a = 6$) they show up.*

Notice that in this case it is also true that it is not just the mode with $\ell = N/2$ in (2.3) that gets a large amplification factor by the scheme. All the neighboring ones are also present. However, now their amplitudes and phases are all correlated because they (mostly) are generated by the corner in the initial data. Thus they interfere with each other in ways subtler than the mere beating observed in the prior example; i.e.: the pattern of grid size oscillations has a clear maximum near the positions of the corners in figure 1.5.

In the next section we will discuss a simple strategy to stabilize numerical schemes, to get rid of numerical oscillations and other undesirable effects. The strategy is based on the introduction of artificial (numerical) dissipation to (selectively) damp the higher modes, without significantly affecting the lower modes (where a consistent scheme should behave properly — see remark 2.4).

[4]When a is large, the corner is very weak and the dominant contribution to the mode amplitudes comes from the smooth part of the initial data (which yields very little amplitude in the high modes).

Remark 2.4 *Finally, going back now to the last paragraph in remark 2.1, consider the behavior of G in (2.4) for k small. Namely*

$$G = 1 \pm i\frac{\Delta t}{\Delta x}k + O\left(\frac{\Delta t}{\Delta x}k^3\right) . \tag{2.10}$$

This should be compared with the behavior of the exact solution for the wave equation (1.1) — see remark 1.1 — which evolves Fourier modes according to the rule

$$u \propto \exp\left\{i\frac{k}{\Delta x}(x_n \pm t_j)\right\} \propto \exp\left\{i\left(kn \pm \frac{\Delta t}{\Delta x}kj\right)\right\} .$$

Thus the exact evolution corresponds to a factor G given by

$$G_{exact} = exp\left(\pm i\frac{\Delta t}{\Delta x}k\right) = 1 \pm i\frac{\Delta t}{\Delta x}k + O\left((\frac{\Delta t}{\Delta x}k)^2\right) . \tag{2.11}$$

This should be compared with (2.10) above. It is clear then that (for k small) G is correct up to small terms in k, which is an alternative way of verifying that the scheme (1.7) is consistent.

3 Numerical Viscosity and Stabilized Scheme.

FILL IN HERE THE GOOD SCHEME EQUATIONS. (3.1)

Notation used for Good Scheme in MatLab: $\eta = (\Delta t/\Delta x)^2$ and $\nu = \Delta t/\Delta x^2$.

Next the figures that go with the good scheme.

4 Reference.

For more information regarding stability of numerical schemes (and many other useful numerical topics) a good all-around practical reference is *Numerical Recipes, The Art of Scientific Computing* by W. H. Press, S. A. Teukolsky, W. T. Vetterling and B. P. Flannery. Cambridge U. Press, New York, 1992.

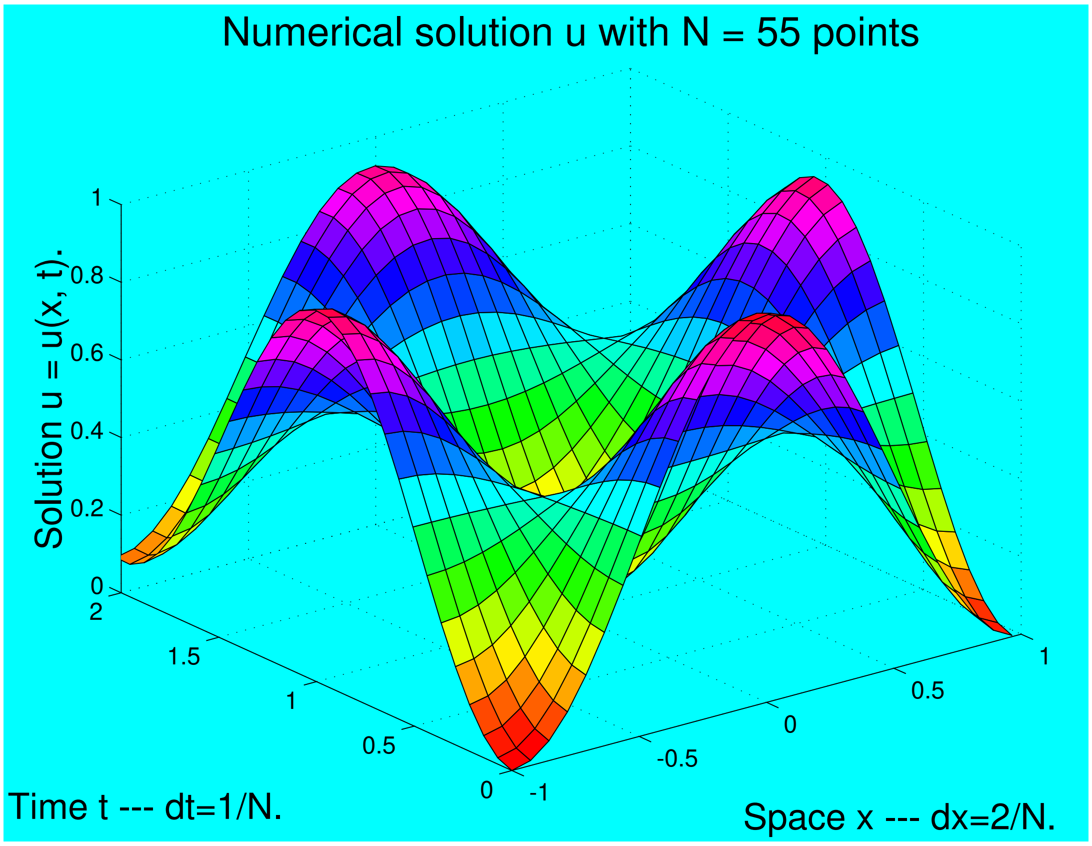

Figure 3.1: Solution of (1.5) with initial data (1.8) using the corrected scheme (3.1) with 55 points in the space grid. To avoid an over-dense graph not all the points in the numerical grid are plotted. However, enough points to show all the relevant details are kept.

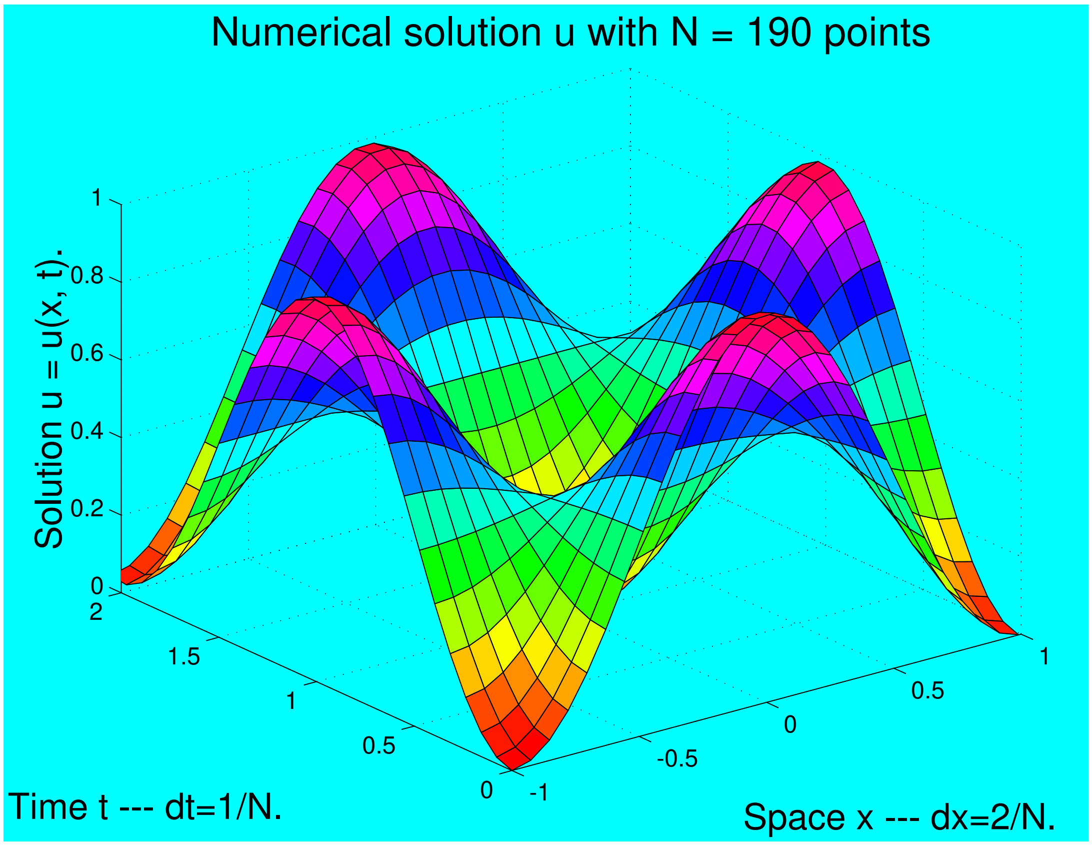

Figure 3.2: Solution of (1.5) with initial data (1.8) using the corrected scheme (3.1) with 190 points in the space grid. To avoid an over-dense graph not all the points in the numerical grid are plotted. However, enough points to show all the relevant details are kept.