Green's functions used as preconditioner

W.D. Drenth, R.M.M. Mattheij and J.M.L. Maubach
Department of Mathematics and Computer Science
Eindhoven University of Technology, The Netherlands

Abstract

The inverse of a boundary value problem operator can be expressed using Green's functions. In the discrete case this is also valid. With this in mind, a preconditioner is constructed using (approximate) Green's functions. The method is described in detail. Subsequently the method is tested for a 1-dimensional problem. For grids with local refinements the results are very good.

1 Introduction

This paper focuses on the use of Green's functions for the construction of preconditioners. As is well known, the inverse of a boundary value problem operator can be represented using Green's functions. This holds for the discrete case as well. Indeed, for solving a linear system of equations (implicit) computation of the inverse of the linear operator is required. For iterative methods a preconditioner is needed, i.e. an approximate inverse. So it makes sense to use (approximate) Green's functions to try and find suitable preconditioners. The preconditioner is constructed in a natural manner as well, as it makes explicitly use of knowledge of the inverse operator, namely the Green's functions.

The idea of using approximate inverses is not entirely new: see [1] and [2] for a recent survey. However, our method differs from the approximate inverse techniques in the following way: In the first place knowledge of the differential operator is used in the construction of the preconditioner. This is most visible in the use of Green's functions. Secondly, the approximate inverse technique makes explicitly use of a prescribed sparsity pattern for the preconditioner

This paper is composed as follows. In section 2 we present some preliminaries and describe the method. Section 3 is devoted to examples in the 1-dimensional case, and in section 4 the 2-dimensional case is briefly presented. We conclude in section 5 with a complexity analysis and final remarks.

2 Preliminaries

Let $\Omega = [x_0, x_1]$ and let p and q be continuous functions. Consider a linear homogeneous selfadjoint second order differential operator

$$\mathcal{L}u = -(pu')' + qu, \tag{1}$$

and assume that $u(x_0) = u(x_1) = 0$. After multiplying $\mathcal{L}u$ by a function $v \in H_0^1(\Omega)$, integrating over Ω and using Green's identities, we have the bilinear form

$$a(u, v) = \int_\Omega pu'v' + quvdx, \ \forall u, v \in H_0^1(\Omega). \tag{2}$$

Furthermore, for $\xi \in \Omega$ define the linear form F_ξ as

$$F_\xi(u) = \int_\Omega \delta_\xi u dx, \ \forall u \in H_0^1(\Omega),$$

where δ_ξ is the Dirac delta function. By property of the delta function, we see that

$$F_\xi(u) = u(\xi),$$

which will be of use later. Note that F_ξ is bounded on Ω and that $a(u, v)$ is bounded and coercive. The conditions for the Lax-Milgram lemma are satisfied. So we may conclude that there exists a $g_\xi \in H_0^1(\Omega)$ such that

$$a(g_\xi, v) = F_\xi(v), \ \forall v \in H_0^1(\Omega). \tag{3}$$

This g_ξ is called the *Green's function* and satisfies the boundary value problem

$$\mathcal{L} g_\xi = \delta_\xi, \tag{4}$$

with homogeneous boundary conditions. More details on Green's function can be found in [3] and [5].

Consider now some discretization of the differential equation in (1) (either by finite elements or finite differences) resulting in an $n \times n$ linear system of equations

$$\mathbf{A}\mathbf{u} = \mathbf{f}. \tag{5}$$

For the construction of a preconditioner $\mathbf{G}$ for $\mathbf{A}$ ($\mathbf{G} \doteq \mathbf{A}^{-1}$) we like to make use of Green's functions in a discrete manner. For some boundary value problems we explicitly know the Green's function. In that situation we can sample the continuous Green's function over the grid points in order to obtain the discrete Green's function. Though, theoretically Green's functions exist, we only have an explicit formula in a few situations. Furthermore, we want to have the property in (4) inherited, i.e.

$$\mathbf{A}\mathbf{g}_i = \mathbf{e}_i, \tag{6}$$

where $\mathbf{g}_i$ denotes the discrete Green's function related to the ith nodal point x_i, and $\mathbf{e}_i$ is the ith unit vector. Obviously, the $\mathbf{g}_i$ are the columns of $\mathbf{G}$. Note that because of discretization errors the discrete Green's function is not necessarily close to the continuous one. Since we are dealing with (6) we may have to stick to discrete Green's functions.

So, to find $\mathbf{g}_i$, in general we will have to solve (6) for each degree of freedom. As solving (6) equals solving the original problem (5), this is not recommendable, also since then we have $\mathbf{G} = \mathbf{A}^{-1}$. Furthermore, since the inverse of a sparse matrix is full in general, $\mathbf{G}$ will be full.

Hence, we aim at a less expensive method for computing $\mathbf{G}$. We still use equation (6), though now restricted to a subset of these equations. Given $\mathbf{x}_i$, the ith degree of freedom, and $p \geq 0$, we look at the subintervals $[x_{i-p}, x_{i+p}]$ for all i, provided they are contained in Ω. Now, instead of solving (6), we will solve

$$\mathbf{A}_i \tilde{\mathbf{g}}_i = \tilde{\mathbf{e}}_i. \tag{7}$$

Here

$$\mathbf{A}_i = \begin{bmatrix} a_{i-p,i-p} & * & * & * & * \\ & \ddots & & & \\ * & * & a_{i,i} & * & * \\ & & & \ddots & \\ * & * & * & * & a_{i+p,i+p} \end{bmatrix},$$

and $\tilde{\mathbf{e}}_i$ is the unit vector of length $2p+1$ with $\tilde{\mathbf{e}}_{i_{p+1}} = 1$. The $\tilde{\mathbf{g}}_i$ thus computed is extended with zeros to fit the original dimension of the problem and to become the ith column of $\mathbf{G}$. For convenience we will call these columns $\mathbf{g}_i$; they can be seen as *local discrete Green's functions.*

If an explicit formula for the Green's functions is available, $\mathbf{G}$ can be constructed by sampling the known Green's function at the points of interest, i.e. in the subinterval $[x_{i-p}, x_{i+p}]$, only. The approximate Green's functions obtained will be called a truncated Green's function. However, using these truncated Green's functions is not recommendable as such a functions may be a bad solution of (6) The $\mathbf{G}$ obtained in this manner is even no longer positive semi-definite. Figures 1 and 2 give a graphical interpretation of the local and truncated global Green's function.

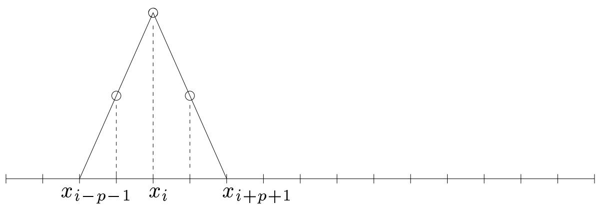

Figure 1: Local Green's function.

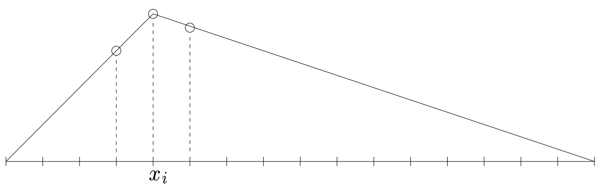

Figure 2: Truncated global Green's function.

3 Green's functions for 1-dimensional diffusion equations

Let $\Omega = (0,1)$. In this section we consider the differential equation

$$\begin{aligned} -u''(x) &= f \quad \text{in } \Omega, \\ u &= 0 \quad \text{on } \partial\Omega. \end{aligned} \tag{8}$$

The Green's functions g_ξ associated to (8) are well known and are given by:

$$g_\xi(x) = \begin{cases} x(1-\xi) & \text{for} \quad x \le \xi, \\ (1-x)\xi & \text{for} \quad x > \xi. \end{cases} \tag{9}$$

We investigate the behaviour of the spectral condition numbers of $\mathbf{A}$ and $\mathbf{AG}$, $\kappa(\mathbf{A})$ and $\kappa(\mathbf{AG})$ respectively, for three different types of grids on Ω. The first grid of interest is an equidistant grid. Secondly we look at two grids with a refinement towards the point 1. The first of these two has a refinement factor of 2, so relatively few points are needed to reach the point 1. The second grid uses a refinement factor of only 1.1, which implies more points are needed to reach the point 1. On the other hand, because of the larger number of degrees of freedom, this grid is more like the equidistant grid and will be better suited to make comparisons. The third grid has refinement areas at 0, 1/2 and 1; this is in fact a 1-dimensional analogue of a brick-mortar configuration; see figure 3. For all three tests, p shown in the first column denotes the radius of the patch on which the local discrete Green's function is defined. The case $p = 0$ corresponds to $\mathbf{G} = \mathbf{I}_n$, so $\kappa(\mathbf{AG}) = \kappa(\mathbf{A})$. The results of the tests are shown in tables 1, 2, 3 and 4. The number of degrees of freedom for the three tests are $2^k - 1$ (equidistant), k (refinement to point 1) and $4k + 3$ (brick-mortar) respectively.

	1 dimension, mesh size (2^{-k})							
p	1/2	1/4	1/8	1/16	1/32	1/64	1/128	1/256
0	1.00	5.83	25.27	103.09	414.35	1659.38	6639.52	26560.07
1	1.00	2.00	6.47	25.61	103.25	414.43	1659.42	6639.54
2	-	1.00	3.19	11.28	45.71	183.89	737.20	2950.55
3	-	1.00	2.00	6.47	25.61	103.26	414.43	1659.42
4	-	-	1.58	4.21	16.36	65.95	265.05	1061.81
5	-	-	1.33	3.19	11.36	45.71	183.91	737.20
6	-	-	1.00	2.49	8.40	33.50	135.00	541.47
7	-	-	1.00	2.00	6.47	25.62	103.26	414.43
8	-	-	-	1.77	5.20	20.20	81.52	327.34
9	-	-	-	1.60	4.29	16.36	65.96	265.05

Table 1: $\kappa(\mathbf{AG})$ for 1 dimensional test problem uniform grid.

The condition number $\kappa(\mathbf{AG})$ satisfies

$$\kappa(\mathbf{AG}) = \frac{\kappa(\mathbf{A})}{(p+1)^2}.$$

This is nicely illustrated by table 1.

	1 dimension, size smallest element (2^{-k})									
p	1/2	1/4	1/8	1/16	1/32	1/64	1/128	1/256	1/512	1/1024
0	1.00	3.87	8.80	18.27	36.93	74.09	148.30	296.67	593.37	1186.75
1	-	1.00	2.36	3.05	4.31	5.25	6.47	7.46	8.66	9.67
2	-	-	1.00	1.95	2.49	2.80	3.54	4.18	4.57	5.25
3	-	-	-	1.00	1.79	2.27	2.55	2.70	3.25	3.78
4	-	-	-	-	1.00	1.71	2.17	2.44	2.58	2.66

Table 2: $\kappa(\mathbf{AG})$ for a grid with refinement toward 1 with refinement factor 2.

	1 dimension, size smallest element $((1.1)^{-k}/10)$, only k is shown								
p	10	20	30	40	50	60	70	80	90
0	127.27	543.12	1571.98	4167.70	10851.19	28162.22	73052.32	189481.55	491467.35
1	28.44	84.77	149.51	216.82	284.82	352.93	420.96	488.87	556.66
2	12.64	37.57	66.31	96.30	126.62	156.97	187.31	217.63	247.86
3	7.13	21.00	37.32	54.22	71.37	88.53	105.71	122.84	139.96
4	4.25	13.40	23.88	34.80	45.85	56.93	68.00	79.05	90.08

Table 3: $\kappa(\mathbf{AG})$ for a grid with refinement toward 1 with refinement factor 1.1.

	1 dimension, mesh size smallest elements (2^{-k})							
p	1/8	1/16	1/32	1/64	1/128	1/256	1/256	1/512
0	25.27	72.57	196.56	500.53	1221.90	2893.33	6696.20	15225.86
1	6.47	16.20	38.63	91.26	205.58	455.37	986.01	2119.01
2	3.19	6.81	15.18	34.15	78.53	174.40	381.15	821.69
3	2.00	4.00	8.82	17.78	40.21	95.18	209.19	443.48
4	-	2.68	5.62	11.22	22.53	49.11	112.86	255.00
5	-	2.00	4.00	8.39	16.34	32.69	68.40	152.46
6	-	-	2.63	5.31	10.95	21.58	43.19	89.36
7	-	-	2.00	4.00	8.00	16.19	32.16	64.32
8	-	-	-	2.63	5.31	10.77	21.47	42.78
9	-	-	-	2.00	4.00	8.00	16.16	32.09
10	-	-	-	-	2.63	5.31	10.66	21.45

Table 4: $\kappa(\mathbf{A})$ for 1 dimensional brick-mortar test problem.

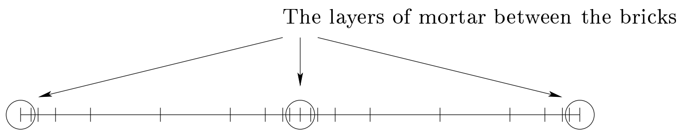

Figure 3: A 1-dimensional brick-mortar configuration.

For the two tests with the refinement to the point 1, tables 2 and 3, we observe an excellent decrease in condition number. As can be expected from the test with refinement factor of 1.1, this gain is not immediately seen as for small k this grid resembles an equidistant grid. The behaviour of $\kappa(\mathbf{AG})$ in the last test is somewhat between the first and second tests: For large mesh sizes the grid is close to an equidistant grid, and for decreasing mesh sizes it becomes more like the grids in the second test.

4 Green's functions in a 2-dimensional setting

In this section we will briefly discuss the above techniques for a 2-dimensional diffusion equation. For the time being we consider only Laplace's equation and an equidistant grid.

The method is similar to that for the 1-dimensional case, as described in section 1. Given a linear system of equations that originates from the underlying partial differential equation (Laplace's equation in our case), we compute the corresponding local discrete Green's function $\mathbf{g}_i$ for each nodal point (source point) $\mathbf{x}_i$ as in (7). These $\mathbf{g}_i$ constitute the matrix $\mathbf{G}$.

As in the 1-dimensional case, let $p \geq 0$ be the radius of the patch on which the local Green's function is to be defined. This means that the best patch would be circular, which is hard to accomplish on a tensor product grid. Henceforth, we look at two possible choices for the patch, which both seem reasonable: a box-type patch, and a star-type patch. Figure 4 illustrates these patches for $p = 2$. For the box-type the number of points is $(2p+1)^2$, which is slightly larger than the $(p+1)^2 + p^2$ points for the star-type. So, we may expect that the box-type to perform a little better.

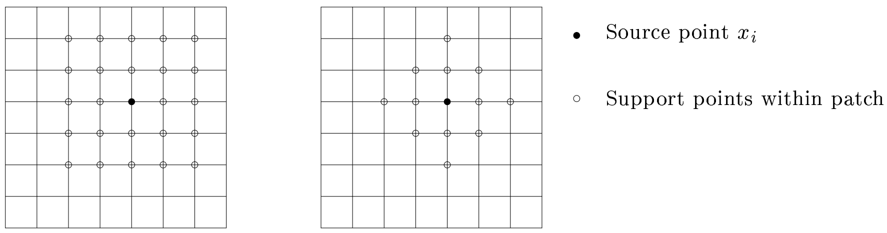

Figure 4: Box-type and star-type patches for $p = 2$.

The results of the tests are shown in tables 5 and 6 for the box-type and star-type patches respectively. We observe a somewhat similar reduction in condition number, as encountered in the 1-dimensional case for an equidistant grid:

$$\kappa(\mathbf{AG}) = C\frac{\kappa\mathbf{A}}{(p+1)^2}.$$

From table 5 it appears that $C \approx \sqrt{2}$.

	2 dimensions, mesh size (2^{-k})				
p	1/2	1/4	1/8	1/16	1/32
0	1.00	5.83	25.27	103.09	414.35
1	1.00	1.52	4.53	17.85	71.48
2	-	1.00	2.15	7.55	29.81
3	-	-	1.48	4.29	16.50
4	-	-	1.24	2.87	10.62
5	-	-	1.11	2.15	7.42
6	-	-	1.00	1.73	5.52
7	-	-	-	1.48	4.30

Table 5: $\kappa(\mathbf{AG})$ for the box-type patch.

	2 dimensions, mesh size (2^{-k})				
p	1/2	1/4	1/8	1/16	1/32
0	1.00	5.83	25.27	103.09	414.35
1	1.00	2.04	6.87	26.30	104.11
2	-	1.35	4.45	18.26	74.06
3	-	1.09	2.24	7.69	30.11
4	-	1.00	1.82	6.71	27.35
5	-	-	1.36	3.79	14.43
6	-	-	1.23	3.48	13.96
7	-	-	1.12	2.32	8.26

Table 6: $\kappa(\mathbf{AG})$ for the star-type patch.

5 Final remarks and conclusions

We start with an $n \times n$ linear system of equations. For each degree of freedom we have to construct the corresponding local discrete Green's function. Let p be the radius of the patch on which the local Green's function has to be defined. Then, for computing each local Green's function, a $(2p+1)\times(2p+1)$ linear system of equations needs to be solved. If this system is solved directly, i.e. by LU decomposition, then the cost will be approximately of $O(8p^3+10p^2)$ for each Green's function. If e.g. $p=1$ then the total computational cost for the construction of $\mathbf{G}$ will be $O(20n)$.
Complexity analysis for the 2-dimensional case can be carried out in a similar way.

Next we will briefly discuss the effect of preconditioning using Green's functions on iterative methods. For simplicity we make comparison between the standard conjugate gradient and the preconditioned conjugate gradient only. The cost for determining $\mathbf{G}$ is approximately 4 times that of CG step only. As is well known, the rate of convergence is determined by the ratio

$$\frac{\kappa(\mathbf{A})-1}{\kappa(\mathbf{A})+1}. \tag{10}$$

This ratio appears in the bound for the norm of the error (see [4]):

$$\mathbf{e}^K \leq \left(\frac{\kappa(\mathbf{A})-1}{\kappa(\mathbf{A})+1}\right)^{2K}\mathbf{e}^0,$$

where $\mathbf{e}^0$ is the initial error, and $\mathbf{e}^K$ is the error at the Kth step of the iterative method.
For the equidistant grid in section 3 we saw a reduction of $\kappa(\mathbf{A})$ by $(p+1)^2$. Is is easily verified that the convergence rate in (10) is replaced by

$$\frac{\kappa(\mathbf{A})-(p+1)}{\kappa(\mathbf{A})+(p+1)}.$$

For small p it is clear the gain in computational speed is visible only for large K, whereas utilizing a large p will require more memory allocation for $\mathbf{G}$ and higher computational cost for constructing $\mathbf{G}$.

However, if we look at the grid with refinement towards 1, and with refinement factor 1.1, we can see a great improvement. For $k = 80$ and $p = 1$, we have

$$\frac{\kappa(\mathbf{A})}{\kappa(\mathbf{AG})} \approx 400.$$

Then (10) reduces by more than 8%. For example, if we would like $\mathbf{e}^K \leq 10^{-4}$, then $K \approx 40$; without preconditioning we would have $K \approx 750$.

In this paper we have constructed a preconditioner based on discrete local Green's functions. For several 1-dimensional examples we have implemented and tested this method. Especially for the grids with local refinements the results are good and promising. In two dimensions a first start has been made, and we hope to see similar results for grids with local refinements as we have seen in the 1-dimensional setting.

References

[1] M. Benzi and M. Tůma. Numerical experiments with two approximate inverse preconditioners. *BIT*, 38:234–241, 1998.

[2] M. Benzi and M. Tůma. A comparative study of sparse approximate inverse preconditioners. *Applied Numerical Mathematics*, 30:305–340, 1999.

[3] R. Courant and D. Hilbert. *Methods of Mathematical Physics, Vol. I.* John Wiley & Sons, New York, 1953.

[4] G. A. Meurant. *Computer solution of the large linear systems.* Elsevier North Holland, Amsterdam, 1999.

[5] G. F. Roach. *Green's Functions. Introductory Theory with Applications.* Van Nostrand Reinhold Company, London, 1970.