Using the program

All computations are done in MATLAB with a few routines written in C in order to speed up the computations. The program requires MATLAB version 7 or later. The program can be down loaded from `http://www.cs.uc.edu/~dschmidt/MATLAB-12-22-2011.zip` as a WinZip file. After extracting all files the first step in MATLAB is to set the path to this directory and to all of its subdirectory. The easiest way of doing this is to use the `set path` command in the `file` menu of MATLAB. First select 'Default' then use 'Add with subfolders...' and finally 'Save'. Intermediate results will be put into the root of directory and it is best to remove all `.mat` files before each run, or to move those which are important to somewhere else.

To get started enter `searchnew` in the command line of MATLAB. It will bring up a window on how to enter the initial matrix. The options are

- *Random:* This will generate a matrix with random entries, its size and type will then be selected in pop-up windows which follow.

- *Load from local computer:* This will allow one to browse the local computer and to select a matrix which was created previously and stored as a MATLAB file with the extension `.mat`. The matrices from `http://www.cs.uc.edu/~dschmidt/PavingTable.htm` are found in the subfolder `bestiary` and can be loaded from there instead from the web. Note that the matrices in `bestiary` are from the time when the zip file was created, and do not reflect any changes since then.

- *Load from web:* **Does not work at this time.** The problem is the interaction of MATLAB with the web.

- *Enter from keyboard:* The matrix has to be entered element by element in the format required by MATLAB. Of course this will be very tedious even for relatively small matrices.

- *Direct Sum:* First select a matrix of size m and then augment it with a zero matrix of size n in order to create a matrix of size $m + n$. This can be used for example when the paving is monotone, as indicated by `"` in the paving table.

- *Tensor Product:* First select a matrix of size m and then take the outer product with an identity matrix of size n in order to create a matrix of size $m \times n$. This is useful for circulant and Toeplitz matrices.

- *Custom construction:* Existing smaller matrices can be used to create a new matrix. After selecting the size of the matrix to be constructed, say n, one then can enter an integer k with $1 \leq k \leq n$. In what follows one has to select via browsing a matrix A of size $\leq (n-k+1)$ so that it will fit as a submatrix when its upper left element is placed on the diagonal at position (k, k). This process continues until 0 is entered. Assuming that the integers $k_1, k_2, \ldots, k_t$ have been entered before 0, and with it the matices $A_1, A_2, \ldots, A_t$ have been selected, then the user is asked to enter an array of real numbers $[c_1, c_2, \ldots, c_t]$ in MATLAB format. With it the matrix $\sum_{i=1}^{t} c_i A_i$ is formed and used as the initial matrix for the search.

The initial matrix is then classified and in another pop up window one has to select which of the possible search spaces to use. This brings up the main window called `Alternate Projection Plotter`. The upper half of it is devoted to provide a graphical representation of the paving numbers, as they are computed. The values are also listed in the MATLAB command window with some additional information. Looking at columns of numbers is usually not as informative as a graph where trends are more easily discerned. Since the axis are computed automatically, one has to look also at the numbers on the axis to obtain a correct picture of what is happening.

The lower half of the window is devoted for entering parameters which control the computation and to provide some information about the computation. Three values for the paving number are displayed `Nmax`, `Nbest` and `Ncur`. The first is the overall maximum value of the current run. The second is the maximum value for the current number of iterations and the last value gives the value which is plotted in the window above.

The important parameters for the alternate projection are `iterations` and `maxradius`. The second parameter determines how far to stretch sub-matrices in order to get better paving numbers. If a matrix is optimal then `maxradius` has to be equal to the paving number, so that `Acur=Amax`. The parameter `iterations` tells how many iterations of the alternate projection program should be carried out.

Note that both parameters can be changed at any time while the program is running. It is important to terminate the entry by pressing the 'Enter' key or the left button, so that the change becomes effective. Setting `iterations` to a value lower then the current number of iterations will terminate the computation at the end of the current iteration. Of course for a large matrix it may take a long time before the current iteration is completed. Nevertheless, the computations are then at a stage, where the work can be continued, whereas by pressing Ctrl-C it is not guaranteed that the matrices can be used.

The slider or the window below it can be used to set by how much the current matrix will be perturbed. If it is set to 1.0 then the current matrix will be replaced completely when `Perturb` is pressed. The radio button `real`, when set, will only perturb the real part of the matrix. When `Gong` is set, it will sound when the simplex method has completed. `Save all` will save intermediate results at the end of each iteration in the root of the directory. The radio button `Grid` is only for the display of the graph and has no significance otherwise.

The top bar has the standard entries from the window operating system plus five additional pull down menus. It is possible to switch between `Packing`, `2-Paving` and `3-Paving` during a run but it is best to stay with one of them unless one knows what relationship exists between the different methods. The top half of each pull down menu selects different versions of the alternate projection method. The different versions only have to do how the projections are generated by the program, and it affects mainly the speed of the computations.

In another part of the pull down menus the simplex method of MATLAB can be selected to find a global maximum of the paving number. MATLAB provides certain default parameters and they are displayed in a pop-up window, but they can be changed to obtain more accurate results. The simplex method can be very time consuming for matrices with lots of independent parameters. This is the case for general matrices even of a small size. For special matrices like circulant ones it may succeed with matrices of intermediate size. It may be necessary to increase the number of iterations and number of function evaluations. The two values appear to be closely related,

so it will be best to change both values. The same applies to the tolerance in the x values and the tolerance in the functional values. Although MATLAB tries to find a global maximum with the help of a simplex search, in most cases the answer will only be a local maximum. If a local maximum is near another one it is worthwhile to repeat the simplex search as another simplex will be used with the new starting value and it is possible to find a larger value.

The pull down menu called `Save` allows the temporary save of up to nine different matrices and then to recall them again during a run. It also allows to set the current matrix to $Amax$, $Abest$ or a perturbed version of these two matrices.

The pull down menu `Save and Quit` allows for the saving of $Amax$, $Abest$ and $Acur$ on a more permanent basis in the computer. `Display` will display in the command window of MATLAB the selected matrix. Finally `Eigenvalues` will plot the eigenvalues of the matrix together with the radius determined by `maxradius`. It is sometimes useful for seeing if one is close to a bad paver or if more improvements can be hoped for.

The program is provided as is, since it is a work in progress. Users are free to modify it. Any questions about the program should be directed to dieter.schmidt$\langle at \rangle$uc.edu