Anymatrix: An Extensible MATLAB Matrix Collection

Nicholas J. Higham and Mantas Mikaitis

Department of Mathematics
University of Manchester
Manchester, UK

7th IMA Conference on Numerical Linear Algebra and Optimization
University of Birmingham, Birmingham, UK
30 June 2022
Slides: `https://bit.ly/3OGt2Bm`

Anymatrix MATLAB toolbox

A **new test matrix collection** in MATLAB and **a tool to collect, search, and share matrices** appended with properties.

Today

Learn about the main features of Anymatrix, how to start using it, and how to make your matrix collections compatible.

Why use Anymatrix for your collections?

- Reproducibility
- Integration in one consistent infrastructure
- Allows to annotate matrices with properties and search by property
- Quick to start in MATLAB

Requirements for a new collection

MATLAB `gallery` does not provide means to search for matrices by their properties and is not customizable by users.

Requirements for a new collection:

- Collect **matrices augmented with properties**.
- **Search for matrices** by their properties.
- **Share collections** in a simple and consistent way.
- Integrate with `git` version control.

Previous work and comparison

Collection	Year	Location	Search by properties	Extensible by user
Regularization Tools	1994	**Own website**	No	No
Matrix Market	1997	**Own website**	Yes	No
`gallery`	2006	**MATLAB**	No	No
CONTEST	2009	**Own website**	No	No
SuiteSparse	2011	**Own website**	Yes	No
NLEVP	2013	GitHub	Yes	No
Matrix Depot	2016	GitHub	No	Yes
IR Tools	2018	**Own website**	No	No
AIR Tools II	2018	**Own website**	No	No
Anymatrix	2021	GitHub	Yes	Yes

Anymatrix groups

```
>> G = anymatrix('groups')
G =
  7×1 cell array
    {'contest' }
    {'core'    }
    {'gallery' }
    {'hadamard'}
    {'matlab'  }
    {'nessie'  }
    {'regtools'}
```

```
>> M = anymatrix('groups', G{2})
M =
  27×1 cell array
    {'core/augment'      }
    {'core/beta'         }
    {'core/biogeography' }
    {'core/blockhouse'   }
    {'core/dembo9'       }
    {'core/edelman27'    }
    {'core/gfpp'         }
    {'core/hess_sublu'   }
    {'core/hessfull01'   }
    {'core/hessmaxdet'   }
    {'core/kms_nonsymm'  }
    {'core/milnes'       }
    {'core/nilpot_triang' }
    {'core/nilpot_tridiag'}
    [...]
```

Matrix IDs

Matrix IDs have the format `<group name>/<matrix name>`.

IDs are unique—this is enforced by the folder structure Anymatrix asks to respect.

Group names are derived from group folders and **matrix names from their `.m` file names**.

Examples

`core/fourier` `regtools/heat` `hadamard/hadamard`

Existent MATLAB matrices

Matrices that come with MATLAB

We have made the gallery and other matrices in MATLAB available through Anymatrix and **documented them with properties**.

```
>> M = anymatrix('groups', 'gallery')
M =
  61×1 cell array
    {'gallery/binomial'   }
    {'gallery/cauchy'     }
    {'gallery/chebspec'   }
    {'gallery/chebvand'   }
    {'gallery/chow'       }
    {'gallery/circul'     }
    {'gallery/clement'    }
    {'gallery/compar'     }
    {'gallery/condex'     }
    {'gallery/cycol'      }
    [...]
```

```
>> M = anymatrix('groups', 'matlab')
M =
  12×1 cell array
    {'matlab/compan'   }
    {'matlab/hadamard' }
    {'matlab/hankel'   }
    {'matlab/hilb'     }
    {'matlab/invhilb'  }
    {'matlab/magic'    }
    {'matlab/pascal'   }
    {'matlab/rosser'   }
    {'matlab/spiral'   }
    {'matlab/toeplitz' }
    {'matlab/vander'   }
    {'matlab/wilkinson'}
```

Generating matrices

A good start is to look at the **help comments of specific matrices**.

```
>> anymatrix('help', 'gallery/wilk')
 wilk   Various specific matrices devised/discussed by Wilkinson.
    [A, b] = GALLERY('wilk',N) is the matrix or system of order N,
    where N is one of the following:

    N = 3: upper triangular system Ux=b. Inaccurate solution.
    N = 4: lower triangular system Lx=b. Ill-conditioned.
    N = 5: HILB{6}(1:5,2:6)*1.8144. Symmetric positive definite.
    N = 21: W21+, tridiagonal. Eigenvalue problem.

>> W = anymatrix('gallery/wilk', 5)
W =
    0.9072    0.6048    0.4536    0.3629    0.3024
    0.6048    0.4536    0.3629    0.3024    0.2592
    0.4536    0.3629    0.3024    0.2592    0.2268
    0.3629    0.3024    0.2592    0.2268    0.2016
    0.3024    0.2592    0.2268    0.2016    0.1814
```

Generating matrices

```
>> anymatrix('help', 'core/beta')
 beta   Symmetric totally positive matrix of integers.
    beta(n) is an n-by-n symmetric totally positive matrix of integers.
    It is also infinitely divisible.
    [A,R] = beta(n) returns both the matrix and its explicitly constructed
    Cholesky factor R.

>> B = anymatrix('core/beta', 4)
B =
     1     2     3     4
     2     6    12    20
     3    12    30    60
     4    20    60   140
```

Listing all matrices

```
>> M = anymatrix('all')
M =
  151×1 cell array
    {'contest/baitsample'       }
    {'contest/curvature'        }
    {'contest/erdrey'           }
    {'contest/geo'              }
    {'contest/gilbert'          }
    {'contest/kleinberg'        }
    {'contest/lap'              }
    {'contest/lockandkey'       }
    {'contest/mht'              }
    {'contest/pagerank'         }
    {'contest/pathlength'       }
    {'contest/pref'             }
    {'contest/renga'            }
    {'contest/rewire'           }
    {'contest/short'            }
    {'contest/smallw'           }
    {'contest/sticky'           }
    {'contest/unisample'        }
    {'core/augment'             }
    {'core/beta'                }
    {'core/biogeography'        }
    {'core/blockhouse'          }
    {'core/collatz'             }
    {'core/dembo9'              }
    {'core/edelman27'           }
    {'core/fourier'             }
    {'core/gfpp'                }
    {'core/hess_sublu'          }
    {'core/hessfull01'          }
    {'core/hessmaxdet'          }
    {'core/kms_nonsymm'         }
    {'core/milnes'              }
    {'core/nilpot_triang'       }
    {'core/nilpot_tridiag'      }
    {'core/pick'                }
    {'core/rschur'              }
    {'core/soules'              }
    {'core/symmstoch'           }
    {'core/totally_nonneg'      }
    {'core/triminsval01'        }
    [...]
```

Hadamard group

Anymatrix includes a **collection of Hadamard matrices** assembled by N. Sloane (`http://neilsloane.com/hadamard/`).

659 Hadmard matrices are available with dimensions up to 428, under one matrix generator `hadamard/hadamard`.

```
% Generate first 8x8 Hadamard matrix
>> anymatrix('hadamard/hadamard', 8, 1)
ans =
     1     1     1     1     1     1     1     1
     1    -1     1    -1     1    -1     1    -1
     1     1    -1    -1     1     1    -1    -1
     1    -1    -1     1     1    -1    -1     1
     1     1     1     1    -1    -1    -1    -1
     1    -1     1    -1    -1     1    -1     1
     1     1    -1    -1    -1    -1     1     1
     1    -1    -1     1    -1     1     1    -1
```

Some **complex hadamard matrices are also available**.

Example: iterating over Hadamard matrices

We will use Anymatrix to iterate over the `hadamard` group and compute the **growth factors for LU factorization**,

$$\rho_n(A) = \frac{\max_{i,j,k} |a_{ij}^{(k)}|}{\max_{i,j} |a_{ij}|}.$$

Here $A \in \mathbb{R}^{n\times n}$, $A^{(1)} = A$ and the elements $a_{ij}^{(k)}$ are the elements at the start of the kth stage of the LU factorization.

Our goal is to test **Cryer's conjecture** (1968) that $\rho_n = n$ for Hadamard matrices.

Example: iterating over Hadamard matrices

```
for j = 1:2
switch j
   case 1, matrix = 'hadamard/hadamard'; str = '';
   case 2, matrix = 'hadamard/complex_hadamard'; str = 'complex';
end

[~,dims] = anymatrix(matrix);
tol = 100*eps;
nn = length(dims);
for i = 1:nn
   n = dims(i,1); m = dims(i,2);
   for k = 1:m
      A = anymatrix(matrix,n,k);
      [L,U,P,Q,rho] = gep(A,'c');
      if abs(rho - n) >= tol*rho
          error('Growth %g for n = %g, matrix %g\n', rho,n,k)
      end
   end
end
end
```

This code finds that growth factor is always very close to n.

Matrix properties

```
>> M = anymatrix('properties', 'core/dembo9')
P =
  9×1 cell array
    {'built-in'       }
    {'fixed size'     }
    {'hankel'         }
    {'indefinite'     }
    {'integer'        }
    {'rank deficient'}
    {'real'           }
    {'square'         }
    {'symmetric'      }

>> P = anymatrix('properties', 'gallery/forsythe')
M =
  6×1 cell array
    {'built-in'            }
    {'eigenvalues'         }
    {'parameter-dependent'}
    {'real'                }
    {'scalable'            }
    {'square'              }
```

List of supported properties

```
>> anymatrix('properties')
ans =
  49×1 cell array
    {'banded'               }
    {'binary'               }
    {'block Toeplitz'       }
    {'built-in'             }
    {'complex'              }
    {'correlation'          }
    {'defective'            }
    {'diagonally dominant'  }
    {'eigenvalues'          }
    {'fixed size'           }
    {'hankel'               }
    {'hermitian'            }
    {'hessenberg'           }
    {'idempotent'           }
    {'ill conditioned'      }
    {'indefinite'           }
    {'infinitely divisible'}
    {'integer'              }
    {'inverse'              }
    {'involutory'           }
    {'M-matrix'             }
    {'nilpotent'            }
    {'nonnegative'          }
    {'normal'               }
    {'orthogonal'           }
    {'parameter-dependent'  }
    {'permutation'          }
    {'positive'             }
    {'positive definite'    }
    {'pseudo-orthogonal'    }
    {'random'               }
    {'rank deficient'       }
    {'real'                 }
    {'rectangular'          }
    {'scalable'             }
    {'singular values'      }
    {'skew-hermitian'       }
    {'skew-symmetric'       }
    {'sparse'               }
    {'square'               }
    {'stochastic'           }
    [...]
```

Matrix .m files

```
>> type private/dembo9
function [A,properties] = dembo9
%DEMBO9  Symmetric Hankel matrix of order 9 and rank 5.
%   DEMBO9 is a symmetric 9-by-9 Hankel matrix with rank 5 arising from
%   a question raised by Amir Dembo, which Guenter Ziegler and Andrew
%   Odlyzko found yielded an incorrect numerical rank when the
%   eigenvalues were computed by EiSPACK running on a VAX machine.

%   Reference:
%   Eric Grosse and Cleve Moler, Underflow Can Hurt, SIAM News 20(6), 1, 1995.

properties = {'hankel', 'symmetric', 'indefinite', 'fixed size', ...
              'rank deficient', 'integer'};

A = [%
-1  1  1 -1 -1  1  1 -1 -1
 1  1 -1 -1  1  1 -1 -1  1
 1 -1 -1  1  1 -1 -1  1  1
-1 -1  1  1 -1 -1  1  1 -1
-1  1  1 -1 -1  1  1 -1 -1
 1  1 -1 -1  1  1 -1 -1  1
 1 -1 -1  1  1 -1 -1  1 -1
-1 -1  1  1 -1 -1  1 -1  1
-1  1  1 -1 -1  1 -1  1  1];
```

Maps of properties

Anymatrix **appends matrices with some of the properties automatically** based on two property maps (customizable).

```
>> type prop_map

function M = prop_map
%PROP_MAP   Lists of properties that map to other properties.
%   PROP_MAP is an n-by-2 cell array in which the first element in a row
%   is a general property to which the more specific properties in the
%   second element of the row are automatically mapped, avoiding the need
%   for the first elements to be specified.

M = {'banded', {'tridiagonal'}
     'binary', {'permutation'}
     'integer', {'binary'}
     'nonnegative', {'binary', 'positive', 'stochastic', ...
                     'totally nonnegative'}
     'orthogonal', {'permutation'}
     'positive', {'totally positive'}
     'symmetric', {'correlation', 'hankel'}
     'positive definite', {'correlation'}
     'totally nonnegative', {'totally positive'}
   };
```

Maps of properties

```
>> type inv_prop_map

function M = inv_prop_map
%INV_PROP_MAP   Lists of properties whose absence implies other properties.
%   INV_PROP_MAP is an n-by-2 cell array in which the first element in
%   a row is a property that is automatically assigned if the
%   properties in the second element of the row are not present.
%   This avoids the need for the (common) properties to be specified.

M = {'real', {'complex'}
     'scalable', {'fixed size'}
     'square', {'rectangular'}
    };
```

Searching the collection by properties

Two ways to search: **by properties** and **by terms** in the `.m` file comments.

```
>> M = anymatrix('properties', 'unimodular')
M =
  2×1 cell array
    {'core/wilson'    }
    {'gallery/dramadah'}
```

Anymatrix also accepts **Boolean expressions**:

```
>> M = anymatrix('properties','tridiagonal and (symmetric or not positive)')
M =
  8×1 cell array
    {'core/biogeography'  }
    {'core/nilpot_tridiag'}
    {'gallery/clement'    }
    {'gallery/dorr'       }
    {'gallery/lesp'       }
    {'gallery/tridiag'    }
    {'gallery/wilk'       }
    {'matlab/wilkinson'   }
```

Searching the collection by terms

```
>> M = anymatrix('lookfor',' zero ')
M =
  12×1 cell array
    {'contest/pathlength'}
    {'core/hessfull01'   }
    {'core/nilpot_triang'}
    {'core/pick'         }
    {'gallery/chow'      }
    {'gallery/clement'   }
    {'gallery/randcolu'  }
    {'gallery/randcorr'  }
    {'gallery/redheff'   }
    {'gallery/smoke'     }
    {'matlab/hankel'     }
    {'matlab/rosser'     }
```

```
>> M = anymatrix('lookfor','wilkinson')
M =
  4×1 cell array
    {'core/gfpp'       }
    {'gallery/triw'    }
    {'gallery/wilk'    }
    {'matlab/wilkinson'}
```

Testing the collection

Anymatrix 1.2 implements **two types of testing**.

- Property testing: **automatically tests some specified properties** of each matrix, where possible.
- **Custom per-group testing**: an example is `hadamard` group, which has a test that checks all matrices are hadamard.

```
>> anymatrix('test', 'hadamard')
Testing   1 Hadamard matrices of dimension  1.
Testing   1 Hadamard matrices of dimension  2.
[...]
All real Hadamard tests passed.
Testing complex Hadamard matrices of dimension  6.
Testing complex Hadamard matrices of dimension  7.
Testing complex Hadamard matrices of dimension 11.
Testing complex Hadamard matrices of dimension 13.
Testing complex Hadamard matrices of dimension  1.
Testing complex Hadamard matrices of dimension  4.
Testing complex Hadamard matrices of dimension 10.
All complex Hadamard tests passed.
```

Extending Anymatrix

There are many ways to **extend Anymatrix**.

- Extend the built-in and remote groups
- Create new groups
- Change default property maps
- Expand the supported set of properties
- Expand the set of tests for properties
- Offer changes to the base collection and the Anymatrix interface
- **Make existent groups compatible and publish**

Preparing remote groups

Annotate M-files with properties, name them (see manual for conventions), place directly in the root of a GitHub repository and add `Contents.m`.

Example published compatible group

Al-Mohy, Higham, and Liu (2022) used various **Anymatrix nonnormal matrices** to **test new matrix cosine algorithms**.

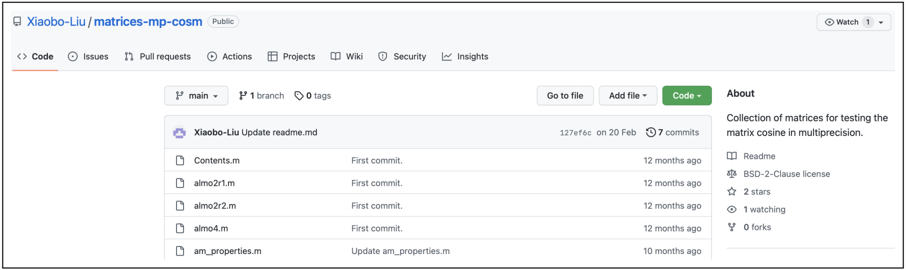
Also collected other matrices from literature in a **new Anymatrix group**.

The group is available online and the matrices used from the base collection can be determined from published scripts.

As a result **their experiments are reproducible easily through Anymatrix**.

Example published compatible group

A group of matrices for testing matrix cosine was published on GitHub by Xiaobo Liu:



The documentation of the group says

Anymatrix Integration

This collection is ready to include as a group in the Anymatrix collection, for which the name `mpcosm` should be used.

Example published compatible group

```
>> anymatrix('groups', 'mpcosm', 'xiaobo-liu/matrices-mp-cosm')
Cloning into '/Users/mantasmikaitis/Work/anymatrix/mpcosm/private'...
remote: Enumerating objects: 58, done.
[...]
Anymatrix remote group cloned.
```

```
>> anymatrix('groups')
ans =
  8×1 cell array
    {'contest' }
    {'core'    }
    {'gallery' }
    {'hadamard'}
    {'matlab'  }
    {'mpcosm'  }
    {'nessie'  }
    {'regtools'}
```

```
>> anymatrix('groups', 'mpcosm')
ans =
  34×1 cell array
    {'mpcosm/almo2r1' }
    {'mpcosm/almo2r2' }
    {'mpcosm/almo4'   }
    {'mpcosm/dahi03'  }
    {'mpcosm/dipa00'  }
    {'mpcosm/edst04'  }
    {'mpcosm/eigt7'   }
    [...]
```

Other compatible groups on GitHub

- 13 **invalid correlation matrices** (https://github.com/higham/matrices-correlation-invalid).
- Square **matrix generator designed for HPL-AI** Benchmark (https://github.com/higham/hpl-ai-matrix).
- Matrices with **specified singular values or condition numbers** (https://github.com/mfasi/randsvdfast-matlab).

```
>> anymatrix('groups')
ans =
  11×1 cell array
    {'contest'      }
    {'core'         }
    {'corrinv'      }
    {'gallery'      }
    {'hadamard'     }
    {'hpl_ai_matrix'}
    {'matlab'       }
    {'mpcosm'       }
    {'nessie'       }
    {'randsvdfast'  }
    {'regtools'     }
```

Shorthand Anymatrix commands

Anymatrix trick

One or two letters of Anymatrix commands are sufficient.

```
>> anymatrix('g')
ans =
  8×1 cell array
    {'contest' }
    {'core'    }
    {'gallery' }
    {'hadamard'}
    {'matlab'  }
    {'mpcosm'  }
    {'nessie'  }
    {'regtools'}

>> anymatrix('h', 'core/dembo9')
 dembo9  Symmetric Hankel matrix of order 9 and rank 5.
    dembo9 is a symmetric 9-by-9 Hankel matrix with rank 5 arising from
    a question raised by Amir Dembo, which Guenter Ziegler and Andrew
    Odlyzko found yielded an incorrect numerical rank when the
    eigenvalues were computed by EiSPACK running on a VAX machine.
```

Conclusion

- Anymatrix **available for free** at `https://github.com/mmikaitis/anymatrix`.
- Anymatrix now at v1.2. We keep adding new matrices and functionality.
- We accept **requests for new additions** or **links to compatible groups**.
- Software can possibly be used for collecting other items in MATLAB.

User's guide: `https://bit.ly/3QgxmsE`.

Paper

N. J. Higham and M. Mikaitis. *Anymatrix: An Extensible MATLAB Matrix Collection*. **Numer. Algorithms., 90:3**. Dec. 2021.
`https://bit.ly/3HeKfzE`.

References I

N. J. Higham and M. Mikaitis.
Anymatrix: an extensible MATLAB matrix collection.
Numer. Algorithms, 90:3. Dec. 2021.

N. J. Higham and M. Mikaitis.
Anymatrix: an extensible MATLAB matrix collection. Users' Guide.
MIMS Eprint 2021.15. Oct. 2021.

A. H. Al-Mohy, N. J. Higham and X. Liu.
Arbitrary precision algorithms for computing the matrix cosine and its Fréchet derivative.
SIAM J. Matrix Anal. Appl., 43:1. Feb. 2022.