MATLAB EXPO 2018

Fit für die MATLAB EXPO

Eine kurze Einführung in MATLAB

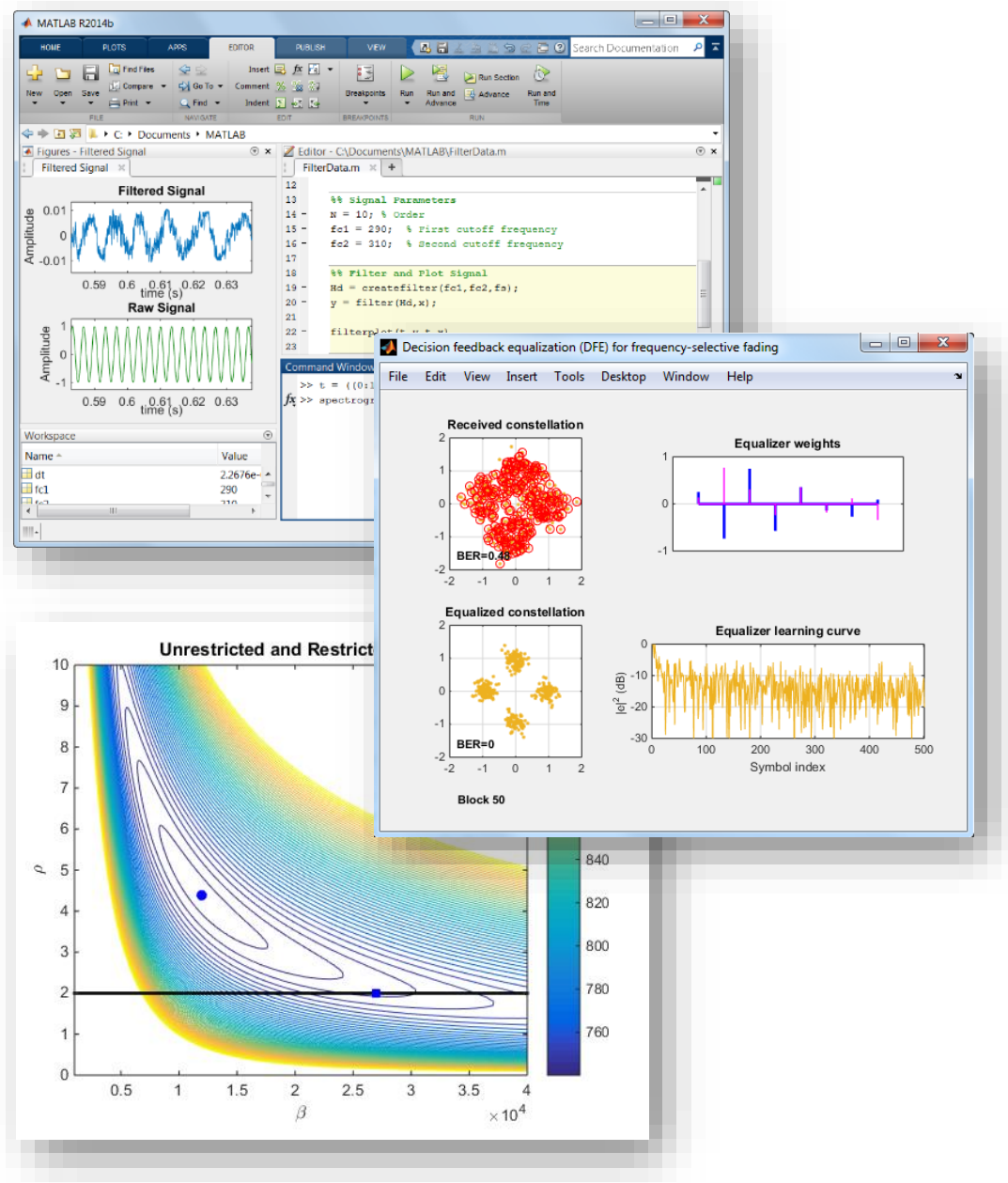
Dr. Jacob Palczynski

Hinweise für Betrachter der PDF Version

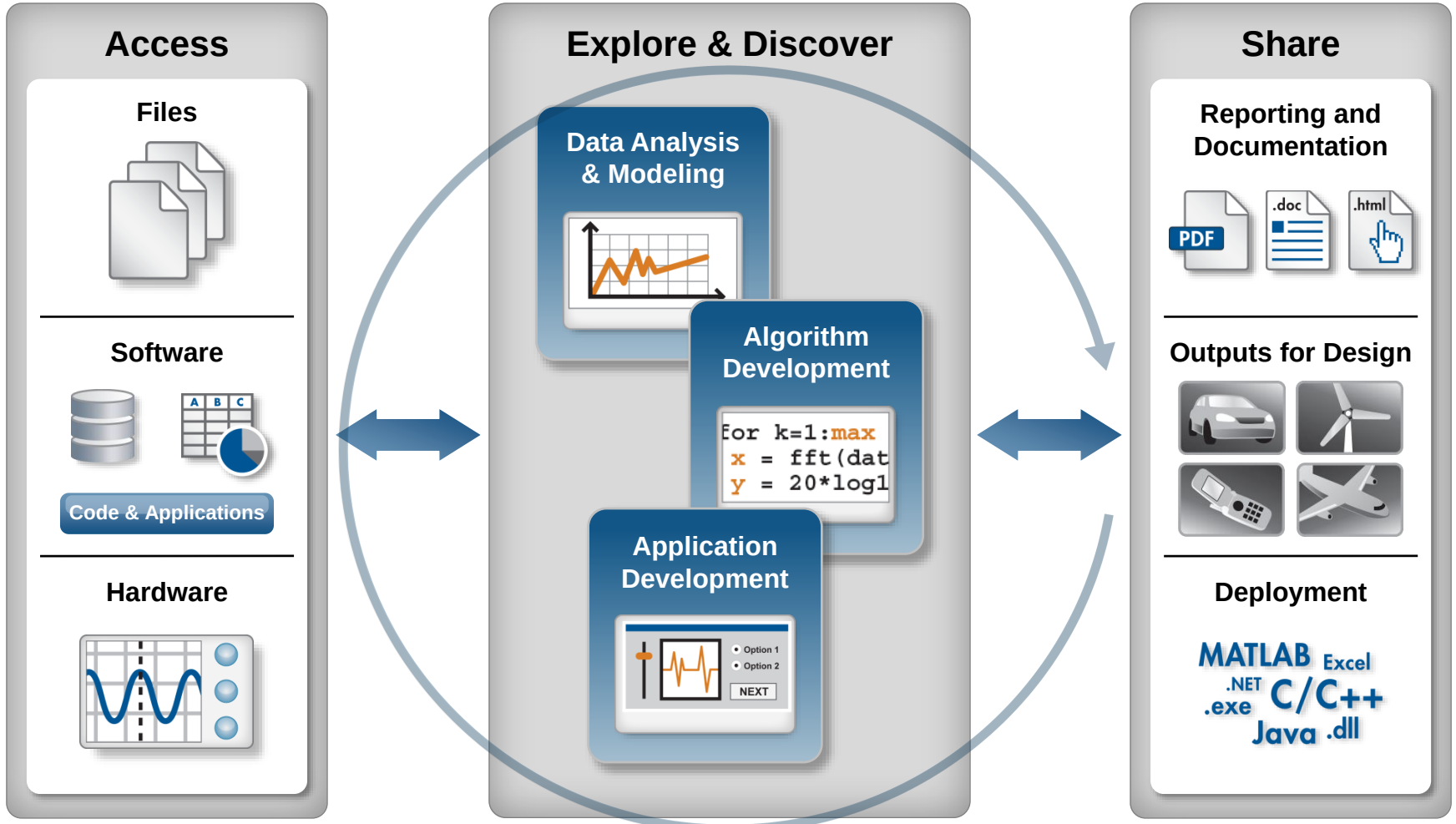
- Die Folien sind eher eine unterstützende Zusammenfassung
- Der Vortrag selbst erfolgt zu großen Teilen direkt in MATLAB
- Das im Vortrag gezeigte Live Script ist angehängt

What is MATLAB?

- High-level language
- Interactive development environment
- Used for:
  - Numerical computation
  - Data analysis and visualization
  - Algorithm development and programming
  - Application development and deployment

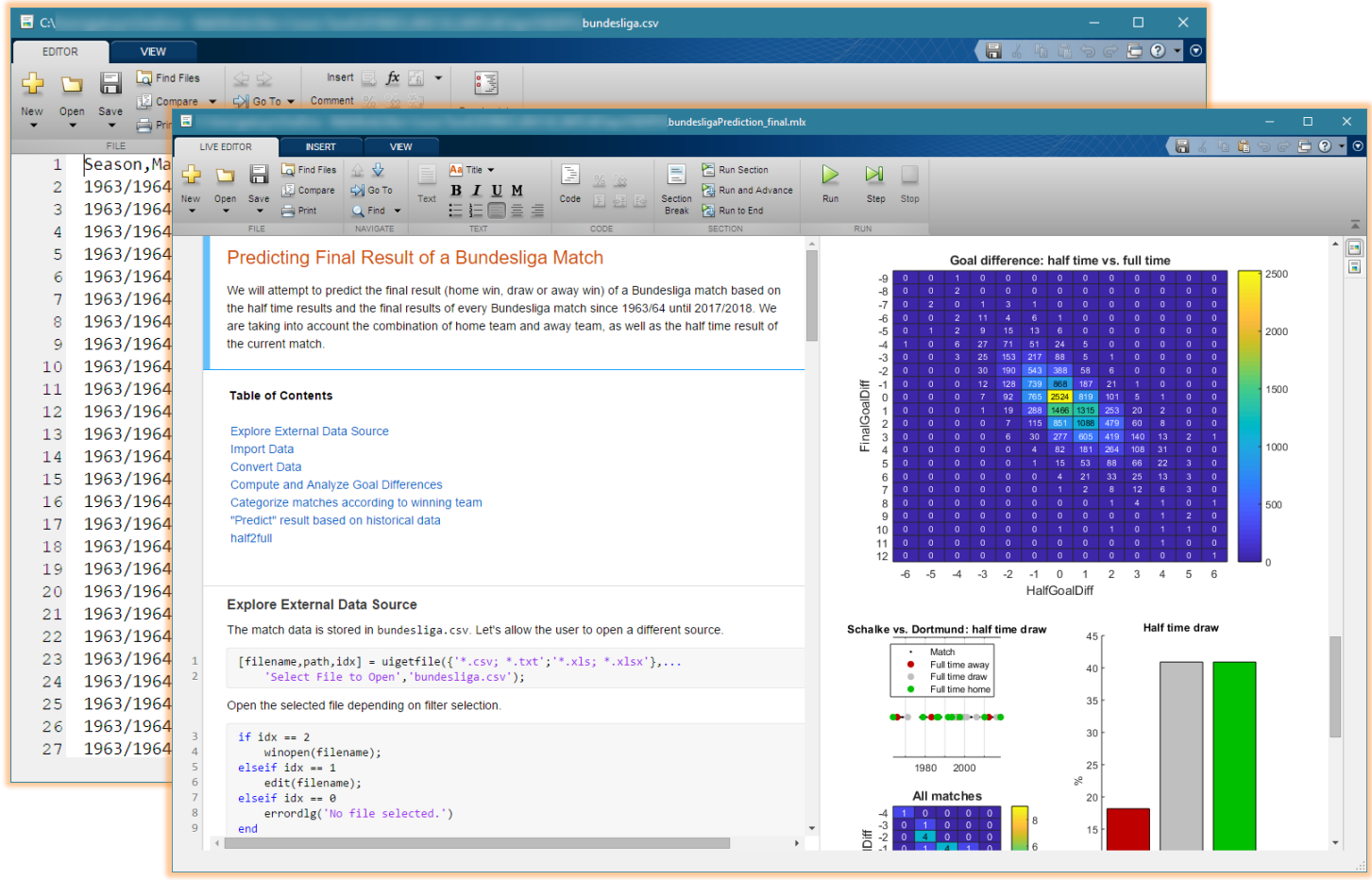


Technical Computing Workflow

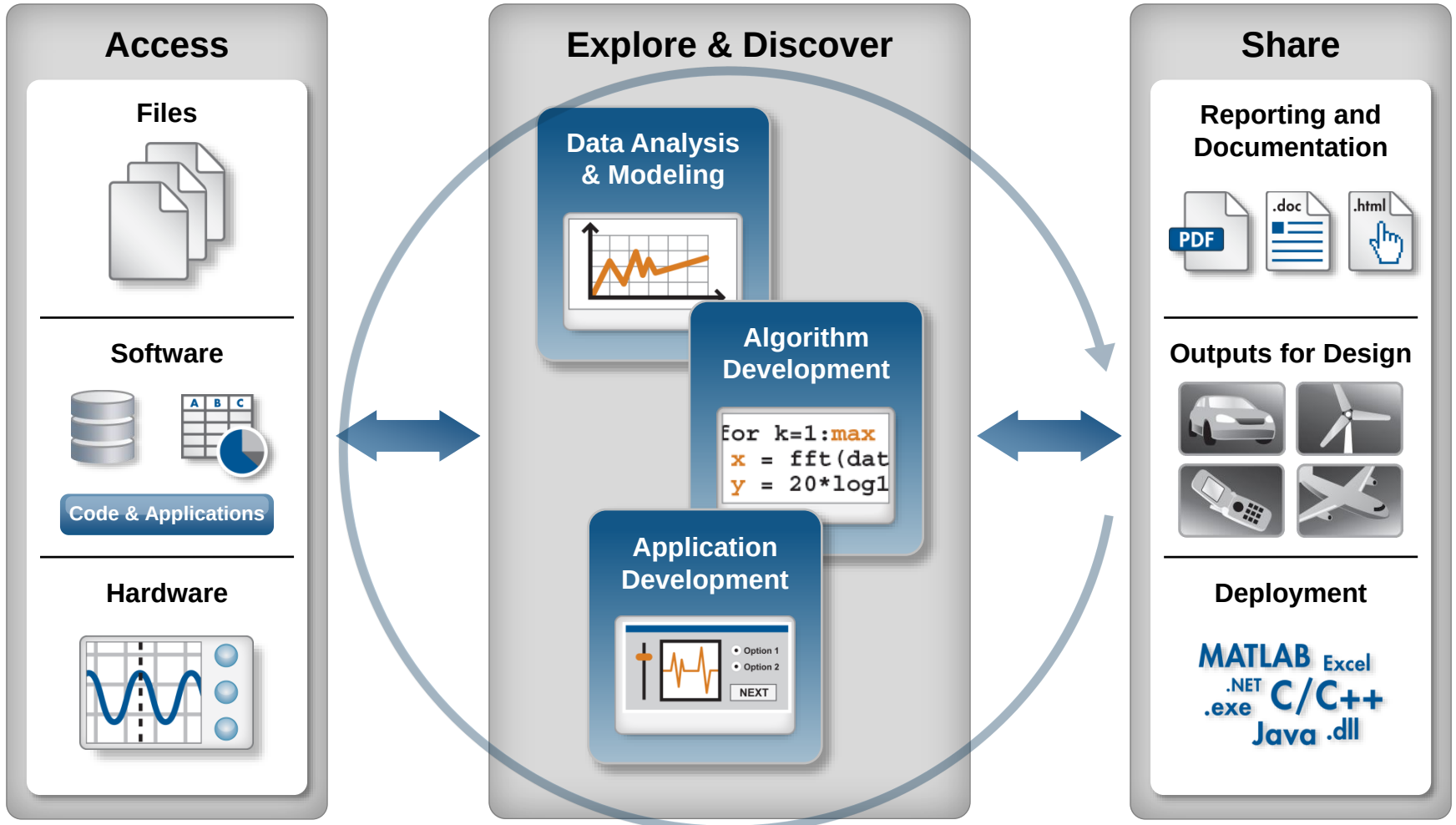


Demo: Analysis of Bundesliga Matches 1963/64 – 2017/2018

- Will the half time “winner“ be the winner after the final whistle?
- Approach:
  - Access data in a csv file
  - Organize data
  - Explore the data interactively
  - Automate analysis
  - Present analysis interactively
  - Document results



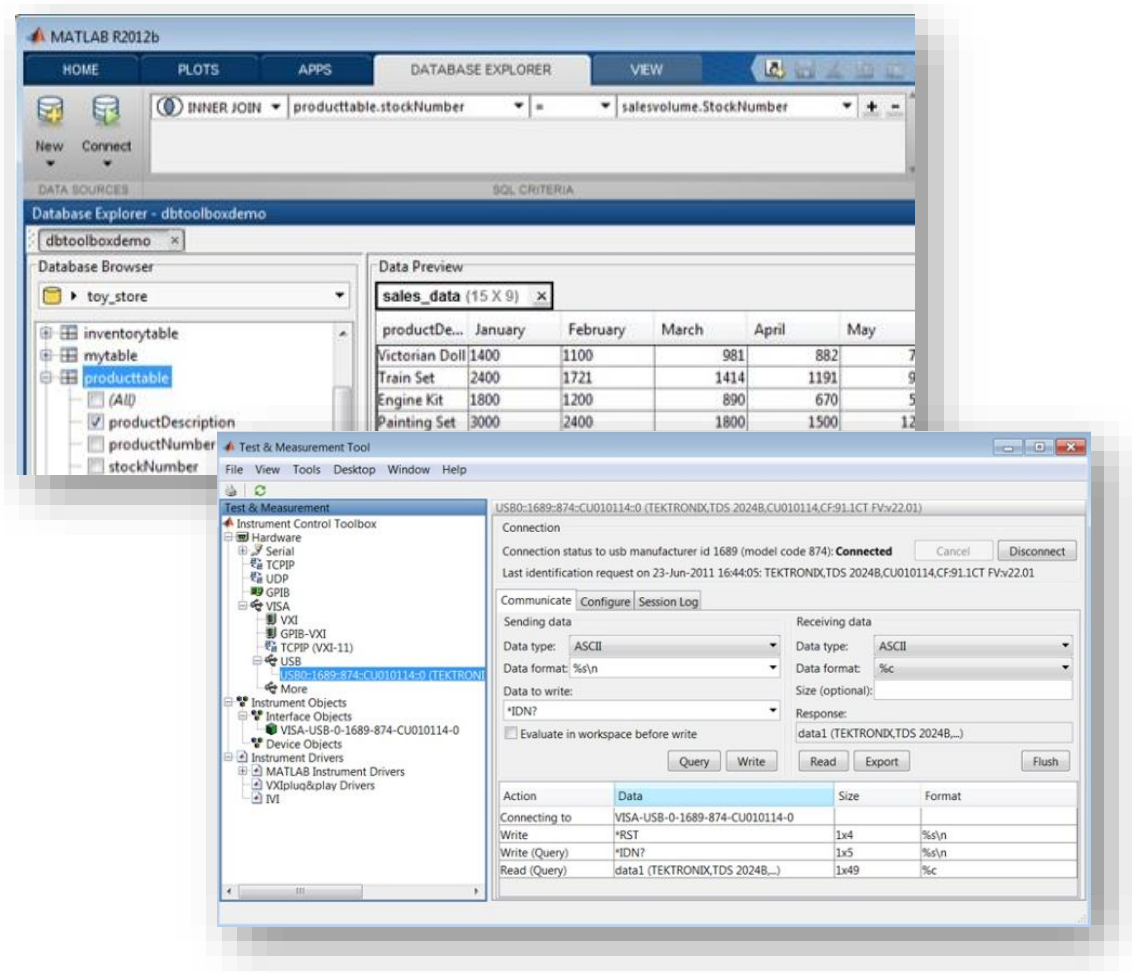
Technical Computing Workflow



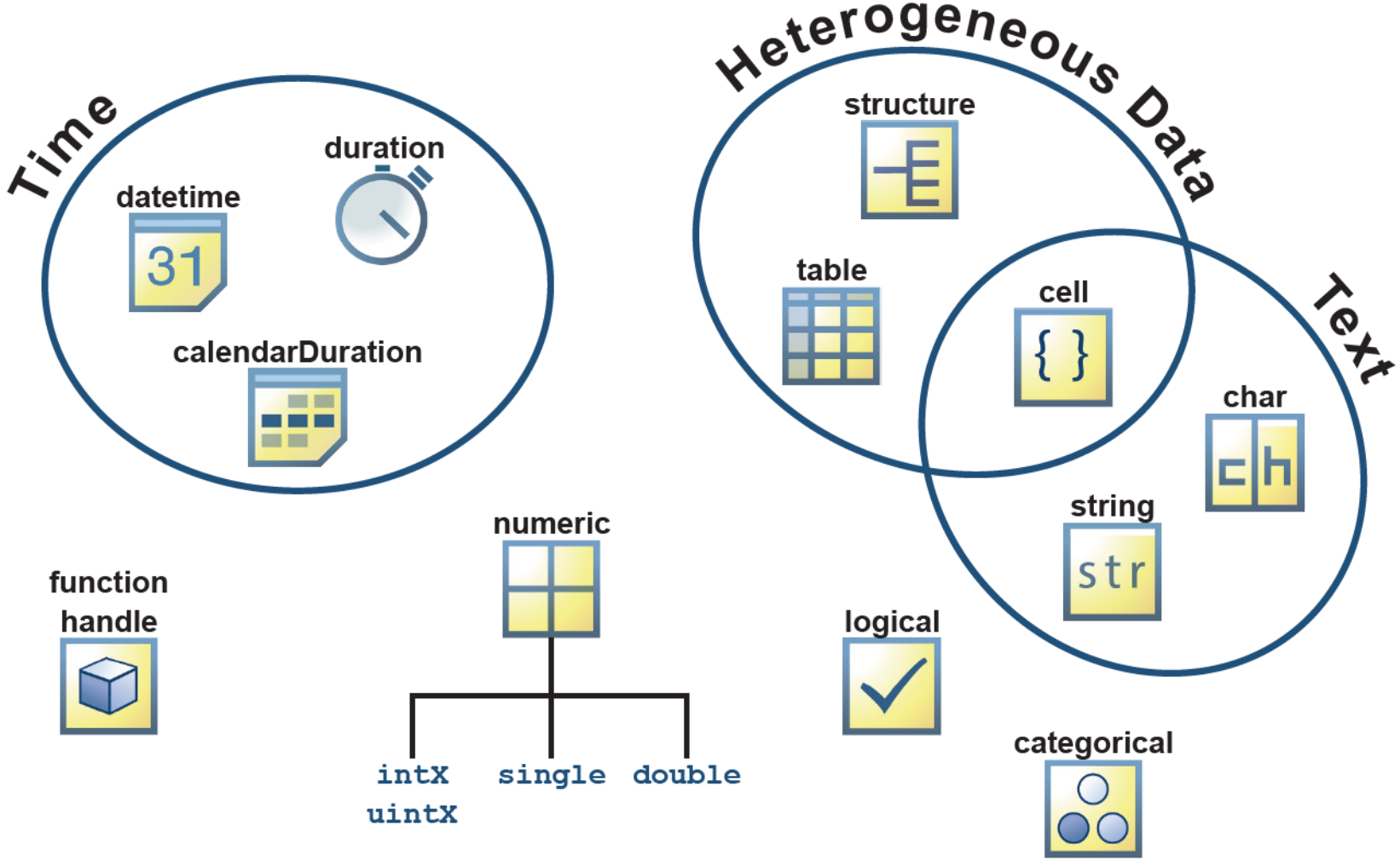
Accessing Data from MATLAB

Access | Explore & Discover | Share

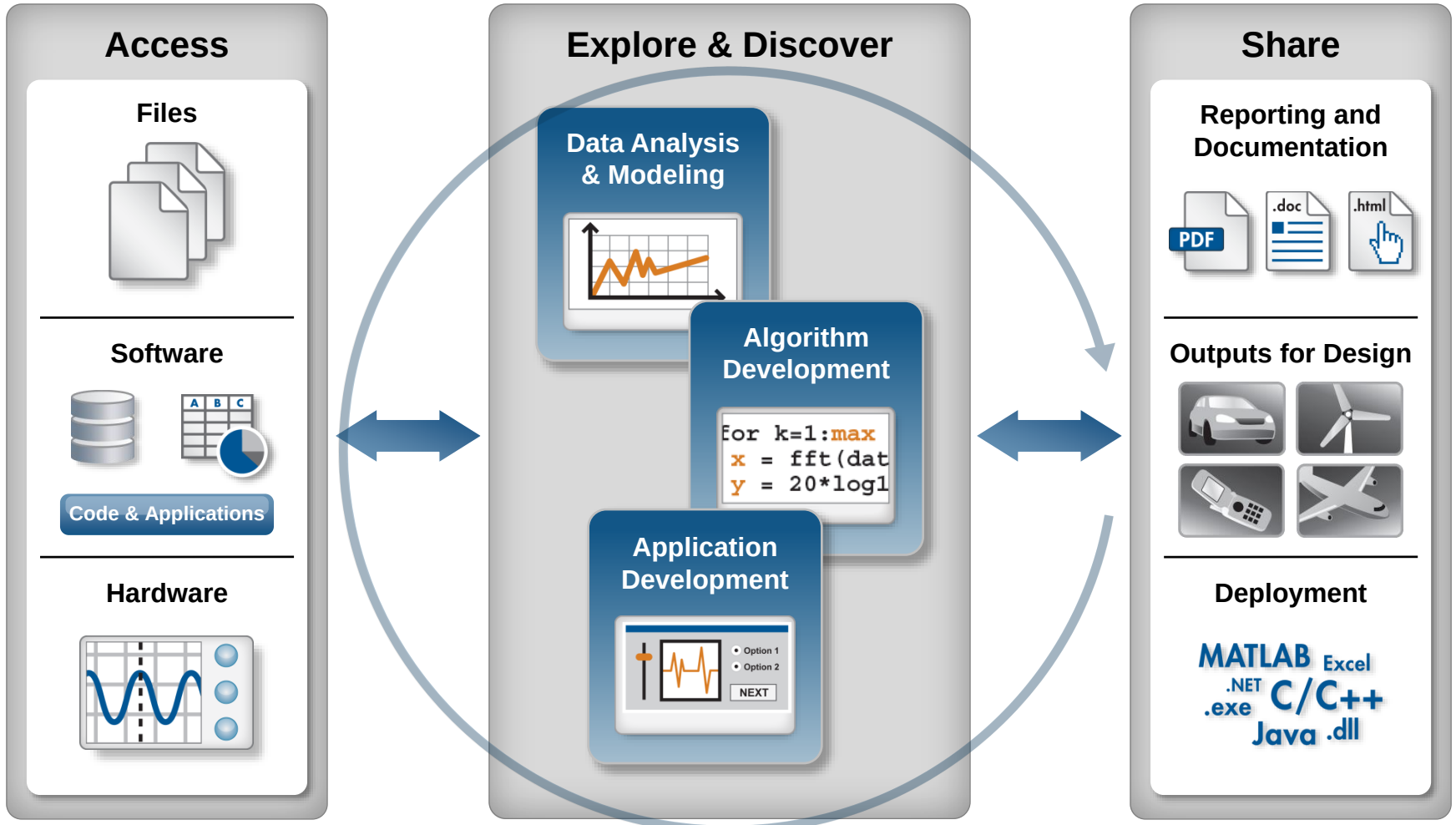
- Files
  - Spreadsheet, text, or binary
  - Audio and video, image
  - Scientific formats and XML
- Applications and languages
  - C/C++, Java, FORTRAN
  - COM, .NET, shared libraries
  - Databases (*Database Toolbox*)
- Measurement hardware
  - Data acquisition hardware *(Data Acquisition Toolbox)*
  - Stand-alone instruments and devices *(Instrument Control Toolbox)*



Organize Data in MATLAB



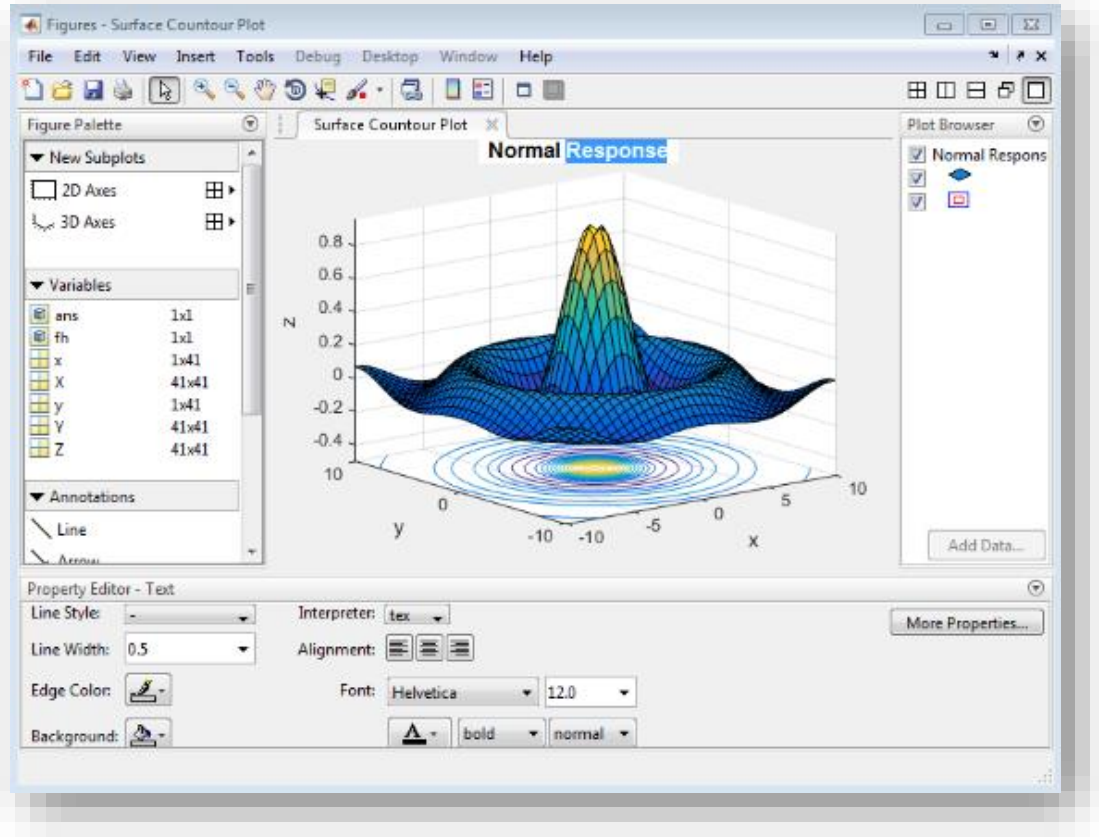
Technical Computing Workflow



Data Analysis and Visualization in MATLAB

Access | **Explore & Discover** | Share

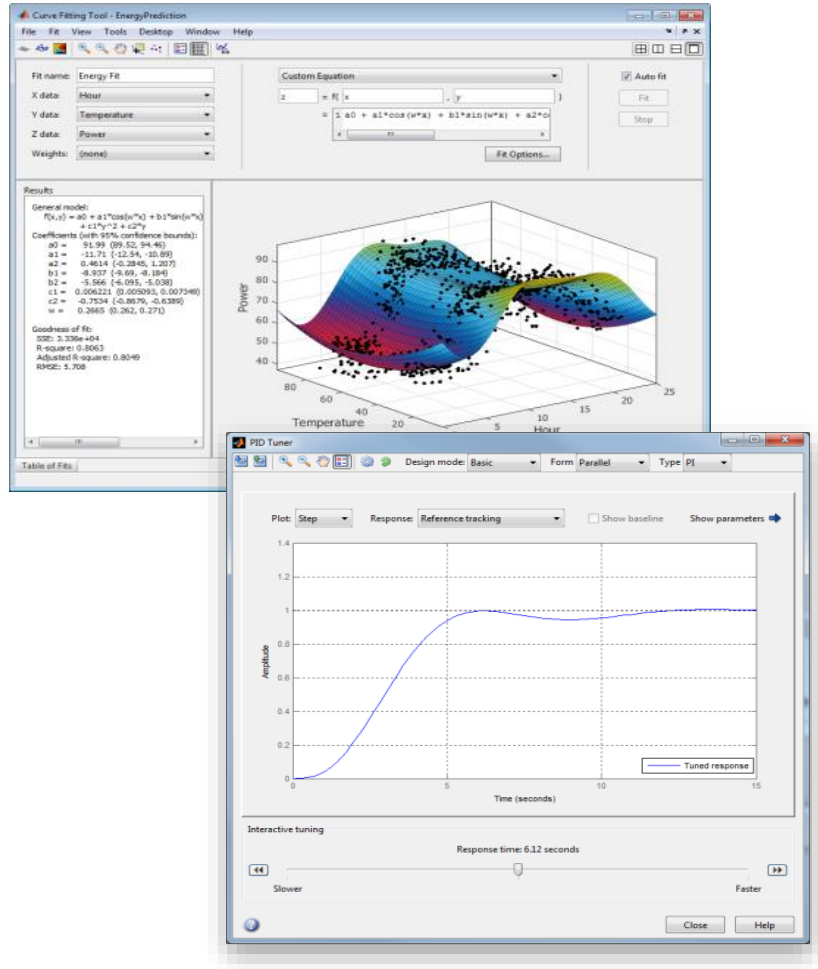
- Built-in engineering and mathematical functions
  - Interpolation, filtering, smoothing, Fourier analysis
- Extensive plotting capabilities
  - 2-D, 3-D, and volume visualization
  - Tools for creating custom plots



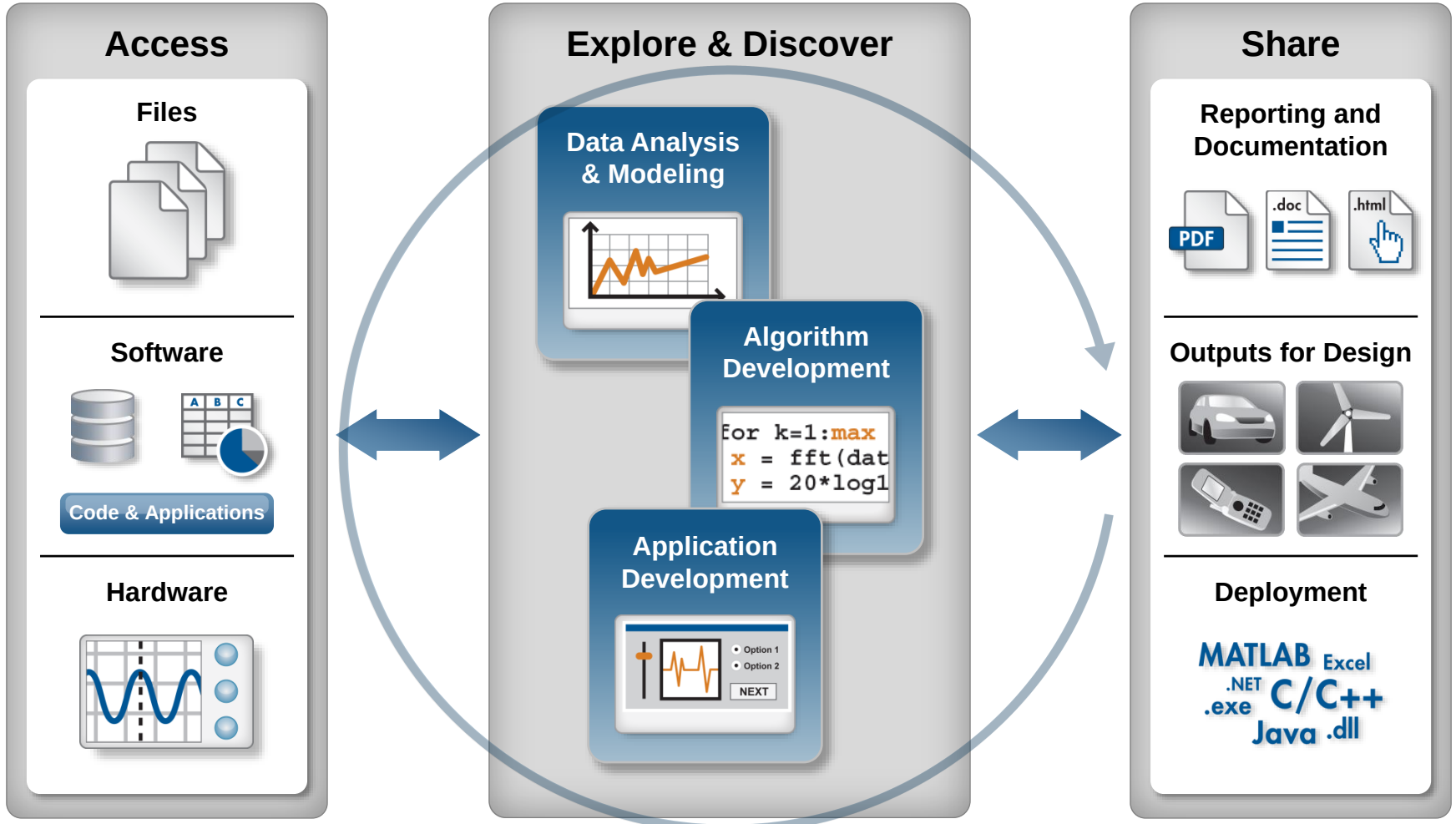
Expanding the Capabilities of MATLAB

Access | **Explore & Discover** | Share

- MathWorks add-on tools for:
  - Math, statistics, and optimization
  - Control system design and analysis
  - Signal processing and communications
  - Image processing and computer vision
  - Parallel computing and more…
- Partner products provide:
  - Additional interfaces
  - Domain-specific analysis
  - Support for niche applications

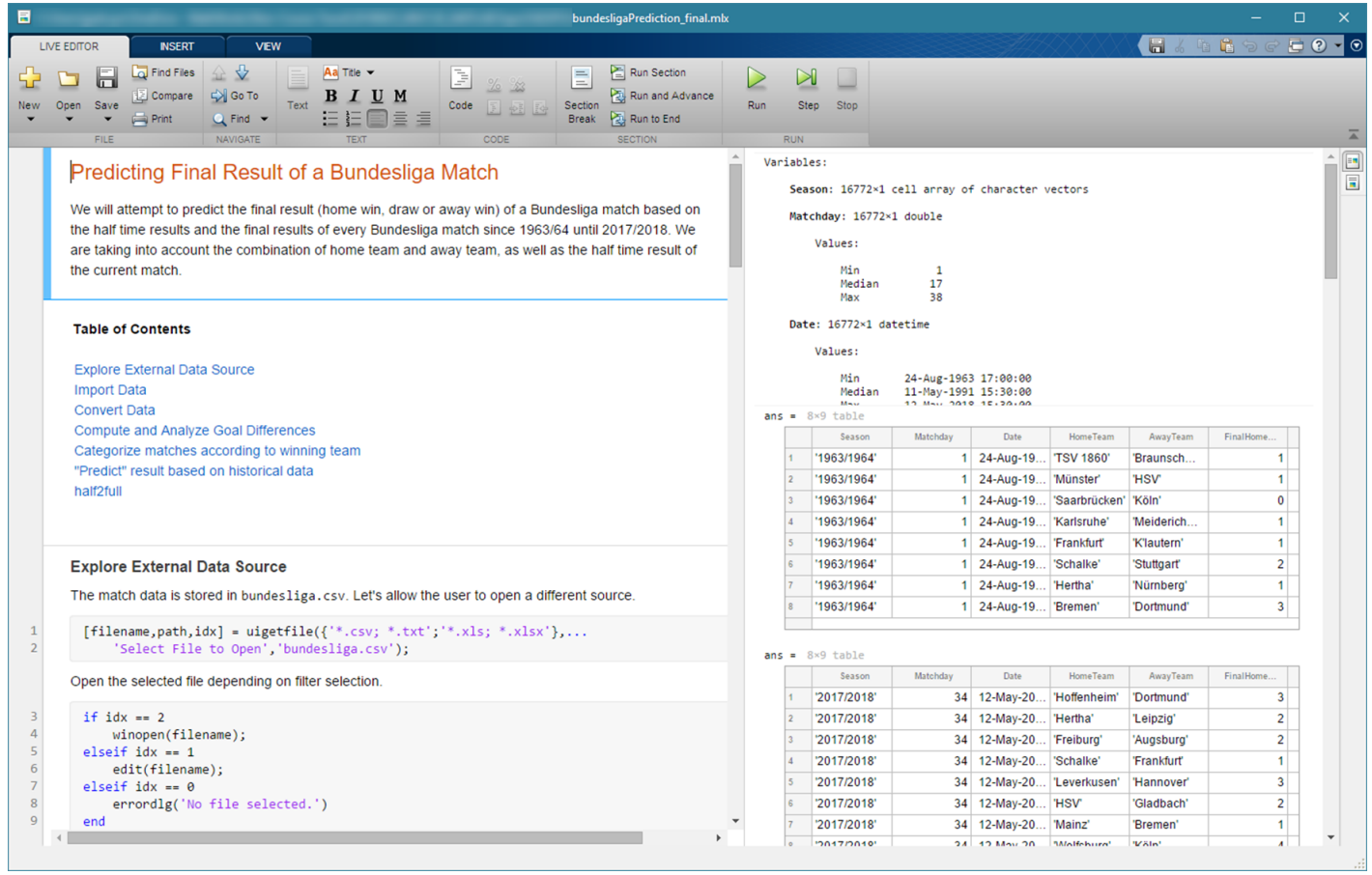


Technical Computing Workflow



Live Scripts – Interactive Documents

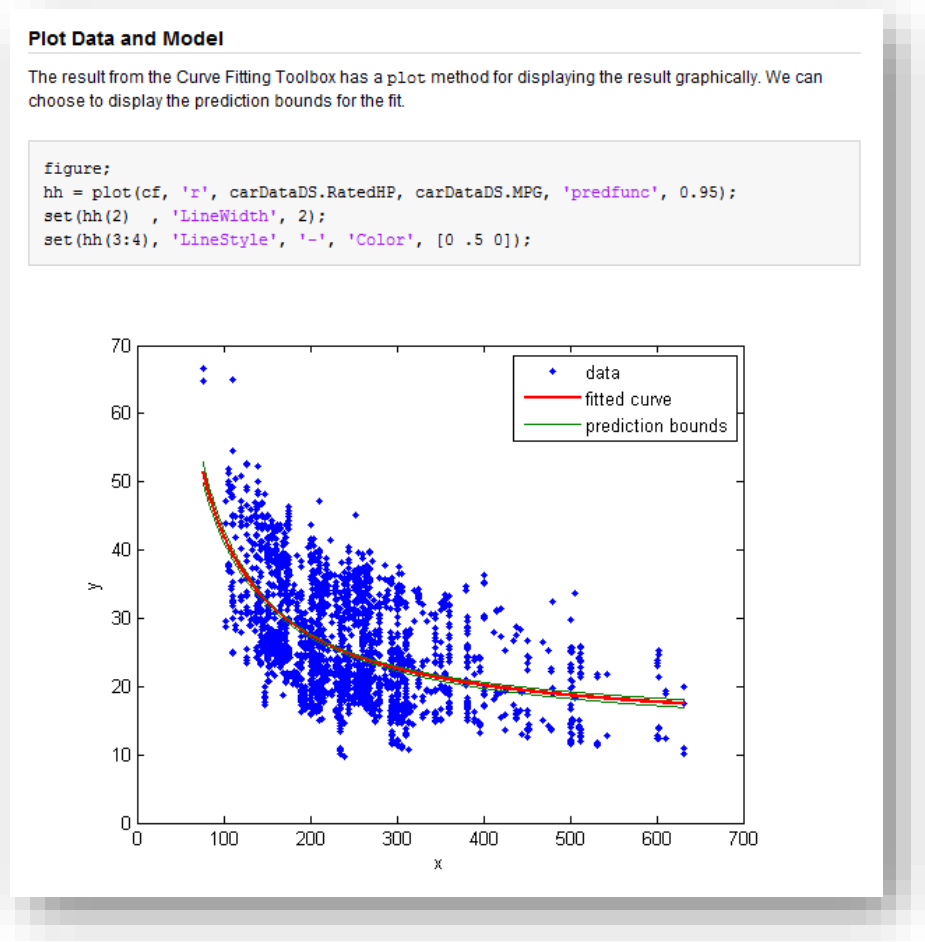
- Structured document
- Executable MATLAB code
- Results
- Formatted text



Sharing Results from MATLAB

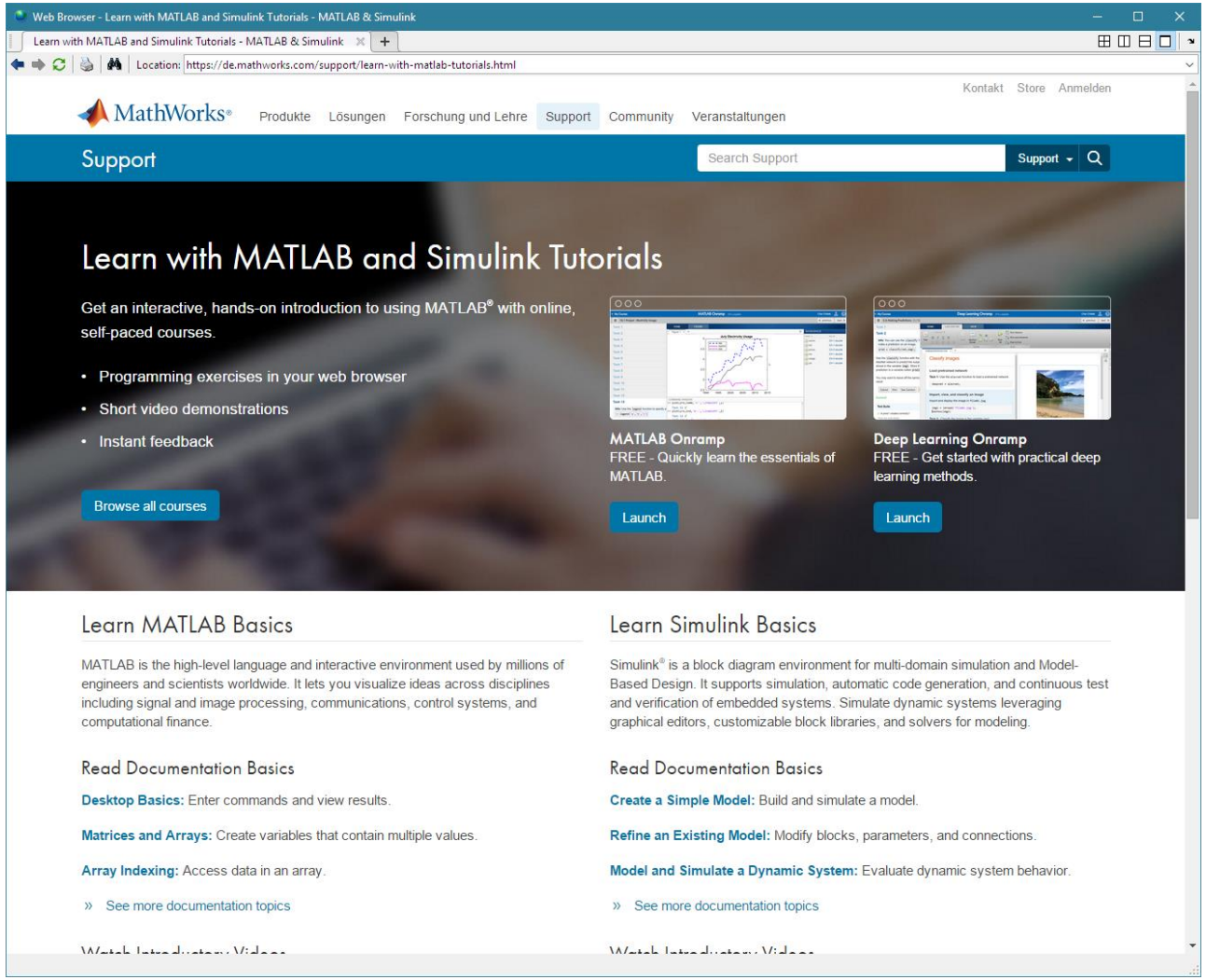
Access | Explore & Discover | **Share**

- Automatically generate reports
  - Save Live Script as PDF, HTML, LaTeX
  - Publish MATLAB files
  - Customize reports using MATLAB Report Generator
- Package as an app or a custom toolbox
- Deploy applications to other environments



Learn more (about) MATLAB

- Documentation
- Webinars
- Training



https://www.mathworks.com/support/learn-with-matlab-tutorials.html

Predicting Final Result of a Bundesliga Match

We will attempt to predict the final result (home win, draw or away win) of a Bundesliga match based on the half time results and the final results of every Bundesliga match since 1963/64 until 2017/2018. We are taking into account the combination of home team and away team, as well as the half time result of the current match.

Table of Contents

Explore External Data Source

The match data is stored in `bundesliga.csv`. Let's allow the user to open a different source.

```
[filename,path,idx] = uigetfile({'*.csv; *.txt';'*.xls; *.xlsx'},...
    'Select File to Open','bundesliga.csv');
```

Open the selected file depending on filter selection.

```
if idx == 2
    winopen(filename);
elseif idx == 1
    edit(filename);
elseif idx == 0
    errordlg('No file selected.')
end
```

Import Data

Import tabular data to a MATLAB table. Every table consist of column variables with variable names. The variables can have different data types, but have to have the same number of elements. We will use the dot notation `x.varname` to access the data stored within the table.

```
liga = readtable(filename);
summary(liga)
```

```
Variables:

    Season: 16772×1 cell array of character vectors

    Matchday: 16772×1 double

        Values:

            Min            1
            Median        17
            Max           38

    Date: 16772×1 datetime

        Values:

            Min       24-Aug-1963 17:00:00
            Median    11-May-1991 15:30:00
            Max       12-May-2018 15:30:00

    HomeTeam: 16772×1 cell array of character vectors

    AwayTeam: 16772×1 cell array of character vectors

    FinalHomeGoals: 16772×1 double

        Values:

            Min               0
            Median            2
            Max              12

    FinalAwayGoals: 16772×1 double

        Values:

            Min              0
            Median           1
            Max              9

    HalfTimeHomeGoals: 16772×1 double

        Values:

            Min               0
            Median            1
            Max               6

    HalfTimeAwayGoals: 16772×1 double

        Values:

            Min               0
            Median            0
            Max               6
```

```
head(liga)
```

```
ans = 8×9 table
                                                                    ...
```

	Season	Matchday	Date	HomeTeam	AwayTeam	FinalHomeGoals
1	'1963/1964'	1	24-Aug-1963...	'TSV 1860'	'Braunschweig'	1
2	'1963/1964'	1	24-Aug-1963...	'Münster'	'HSV'	1
3	'1963/1964'	1	24-Aug-1963...	'Saarbrücken'	'Köln'	0
4	'1963/1964'	1	24-Aug-1963...	'Karlsruhe'	'Meideriche...	1
5	'1963/1964'	1	24-Aug-1963...	'Frankfurt'	'K'lautern'	1
6	'1963/1964'	1	24-Aug-1963...	'Schalke'	'Stuttgart'	2
7	'1963/1964'	1	24-Aug-1963...	'Hertha'	'Nürnberg'	1
8	'1963/1964'	1	24-Aug-1963...	'Bremen'	'Dortmund'	3

```
tail(liga)
```

```
ans = 8×9 table
```

...

	Season	Matchday	Date	HomeTeam	AwayTeam	FinalHomeGoals
1	'2017/2018'	34	12-May-2018...	'Hoffenheim'	'Dortmund'	3
2	'2017/2018'	34	12-May-2018...	'Hertha'	'Leipzig'	2
3	'2017/2018'	34	12-May-2018...	'Freiburg'	'Augsburg'	2
4	'2017/2018'	34	12-May-2018...	'Schalke'	'Frankfurt'	1
5	'2017/2018'	34	12-May-2018...	'Leverkusen'	'Hannover'	3
6	'2017/2018'	34	12-May-2018...	'HSV'	'Gladbach'	2
7	'2017/2018'	34	12-May-2018...	'Mainz'	'Bremen'	1
8	'2017/2018'	34	12-May-2018...	'Wolfsburg'	'Köln'	4

The `readtable` function recognises time representation automatically. Try also

```
liga.Date(end) - liga.Date(1)
```

Convert Data

For efficiency and ease of use, it makes sense to treat textual information as categories. Try also:

```
hometeam = liga.HomeTeam;
nnz(hometeam == 'Schalke')
hometeam_cat = categorical(hometeam);
nnz(hometeam_cat == 'Schalke')
whos hometeam*
```

```
liga.Season = categorical(liga.Season);
```

```
liga.HomeTeam = categorical(liga.HomeTeam);
liga.AwayTeam = categorical(liga.AwayTeam);
```

Meidericher SV got renamed to MSV Duisburg (Duisburg) on 07-Jan-1967 (Wikipedia | MSV Duisburg).

```
liga.HomeTeam = mergecats(liga.HomeTeam,{'Meidericher SV','Duisburg'},'MSV Duisburg');
liga.AwayTeam = mergecats(liga.AwayTeam,{'Meidericher SV','Duisburg'},'MSV Duisburg');
```

Explore team names.

```
team = categories(liga.HomeTeam)
```

```
team = 54×1 cell array
    {'Aachen'           }
    {'Augsburg'         }
    {'Bayern'           }
    {'Bielefeld'        }
    {'Blau-Weiß 90 Ber.'}
    {'Bochum'           }
    {'Braunschweig'     }
    {'Bremen'           }
    {'Cottbus'          }
    {'Darmstadt'        }
    {'Dortmund'         }
    {'Dresden'          }
    {'MSV Duisburg'     }
    {'Düsseldorf'       }
    {'F. Köln'          }
    {'Frankfurt'        }
    {'Freiburg'         }
    {'Fürth'            }
    {'Gladbach'         }
    {'HSV'              }
    {'Haching'          }
    {'Hannover'         }
    {'Hertha'           }
    {'Hoffenheim'       }
    {'Homburg'          }
    {'Ingolstadt'       }
    {'K'lautern'        }
    {'Karlsruhe'        }
    {'Köln'             }
    {'Leipzig'          }
    {'Leverkusen'       }
    {'Mainz'            }
    {'Münster'          }
    {'Neunkirchen'      }
    {'Nürnberg'         }
    {'Oberhausen'       }
    {'Offenbach'        }
    {'Paderborn'        }
    {'RW Essen'         }
    {'Rostock'          }
    {'Saarbrücken'      }
    {'Schalke'          }
    {'St. Pauli'        }
    {'Stuttg. Kick.'    }
    {'Stuttgart'        }
    {'TSV 1860'         }
    {'Tasmania'         }
    {'TeBe Berlin'      }
```

```
    {'Uerdingen'        }
    {'Ulm'              }
    {'Waldhof'          }
    {'Wattenscheid'     }
    {'Wolfsburg'        }
    {'Wuppertaler SV'   }
```

```
isequal(team,categories(liga.AwayTeam))
```

```
ans = logical
   1
```

```
figure(1)
wordcloud(liga{:,{'HomeTeam','AwayTeam'}});
```

Compute and Analyze Goal Differences

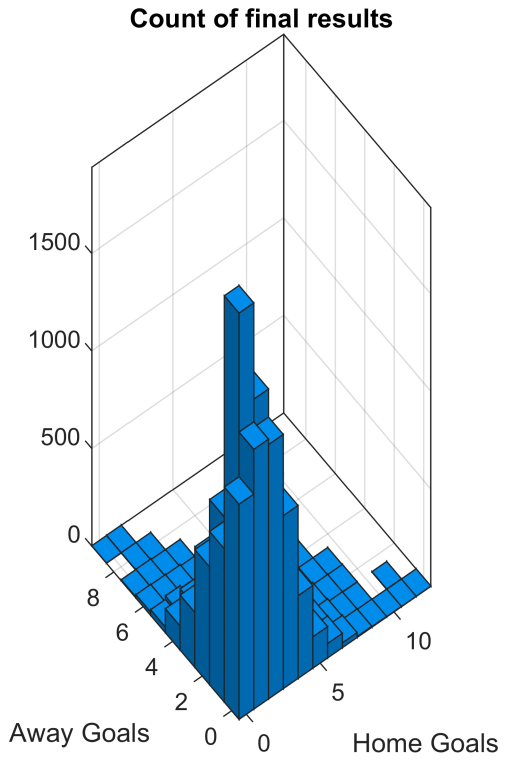
Computing with vectors of numbers is straightforward.

```
liga.HalfGoalDiff = liga.HalfTimeHomeGoals - liga.HalfTimeAwayGoals;
liga.FinalGoalDiff = liga.FinalHomeGoals - liga.FinalAwayGoals;
```

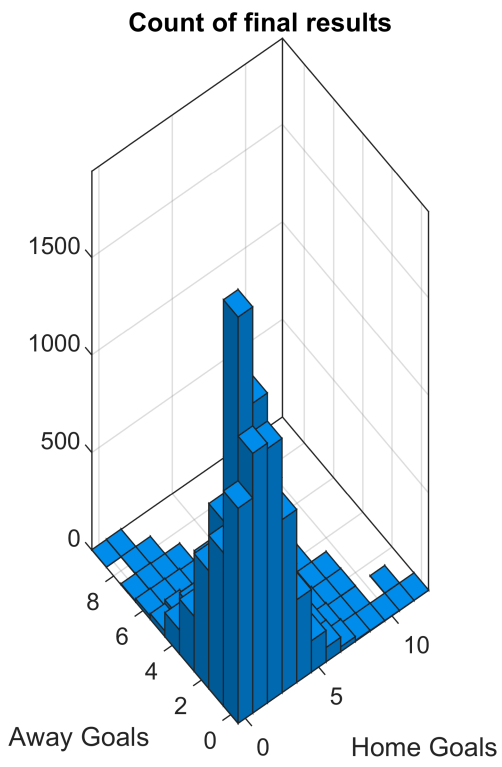
Explore data visually in different ways.

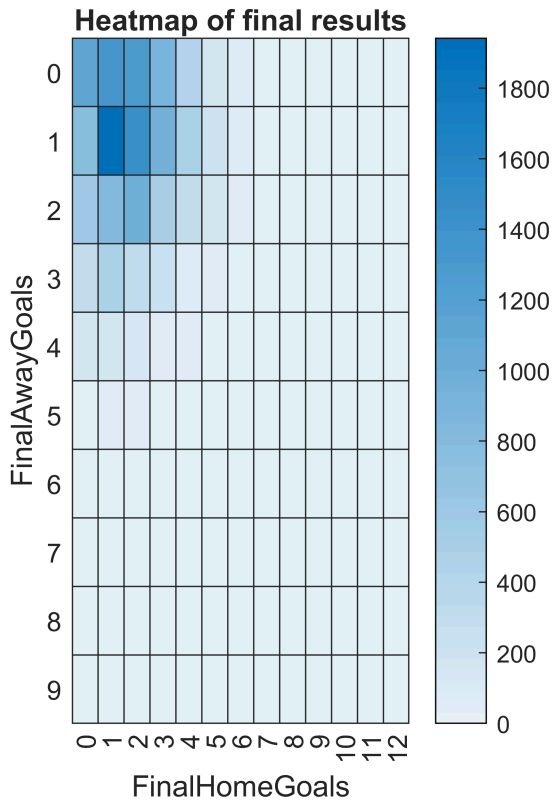
```
figure(2)
subplot(1,2,1)
```

```
histogram2(liga.FinalHomeGoals,liga.FinalAwayGoals)
title("Count of final results")
xlabel("Home Goals")
ylabel("Away Goals")
```

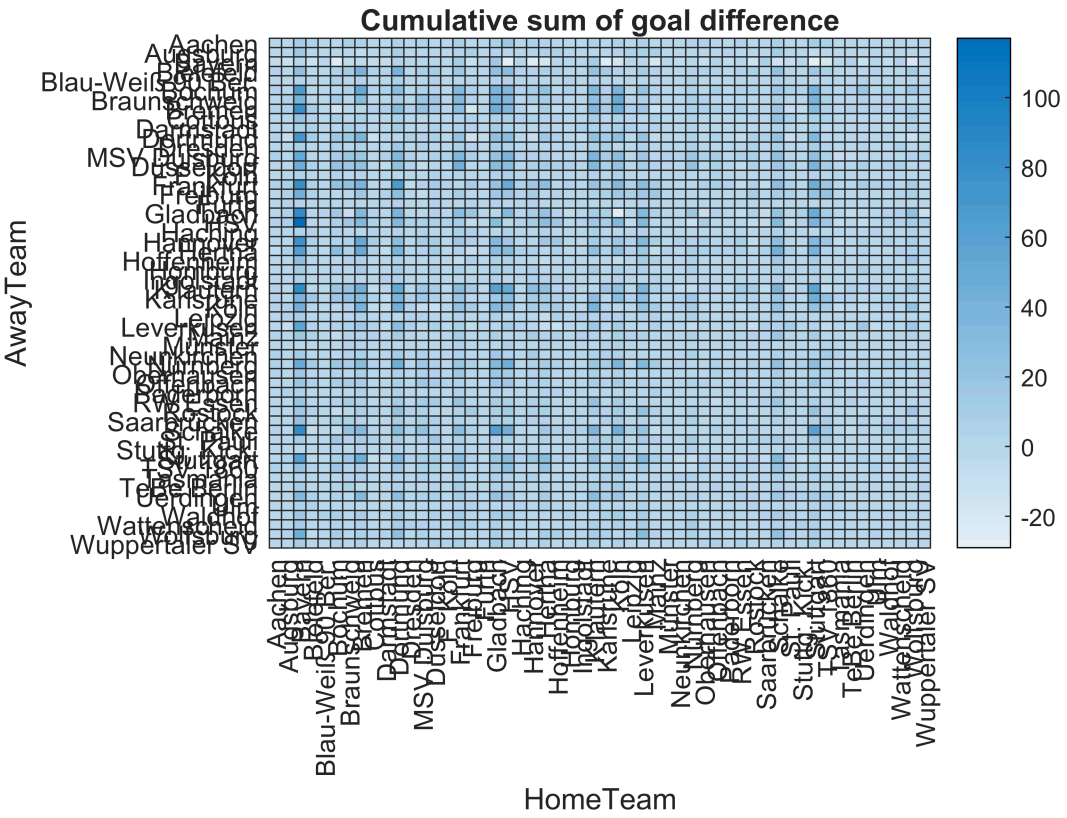



```
subplot(1,2,2)
heatmap(liga,'FinalHomeGoals','FinalAwayGoals',...
    'Title','Heatmap of final results');
```

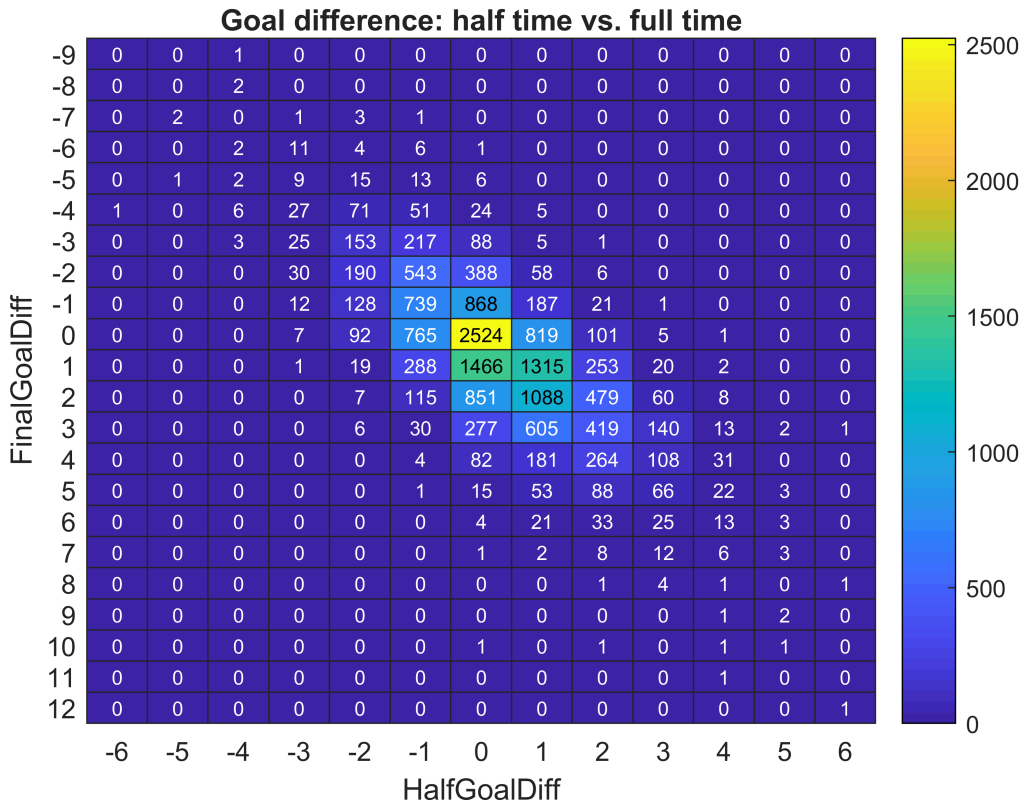





```
figure(3)
heatmap(liga,'HomeTeam','AwayTeam',...
    'ColorVariable','FinalGoalDiff',...
    'ColorMethod','sum',...
    'Title','Cumulative sum of goal difference');
```




```
figure(4)
heatmap(liga,'HalfGoalDiff','FinalGoalDiff',...
    'Colormap',parula,...
    'Title','Goal difference: half time vs. full time');
```



Categorize matches according to winning team

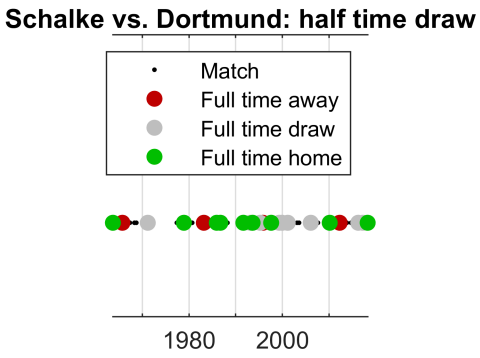
Discretize data to categories `'away'`, `'draw'`, and `'home'`, representing the winning team at half and full time.

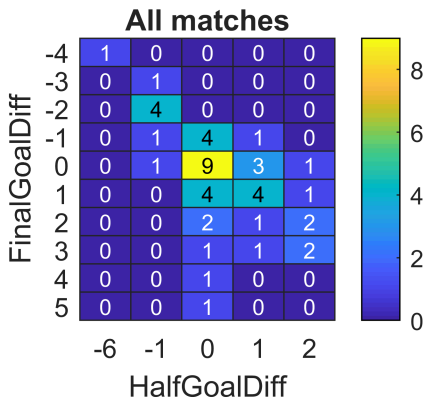
```
resultEdges = [-inf 0 1 inf];
liga.FinalWinner = discretize(liga.FinalGoalDiff,resultEdges,...
    'categorical',{'away','draw','home'});
liga.HalfTimeWinner = discretize(sign(liga.HalfGoalDiff),resultEdges,...
    'categorical',{'away','draw','home'});
```

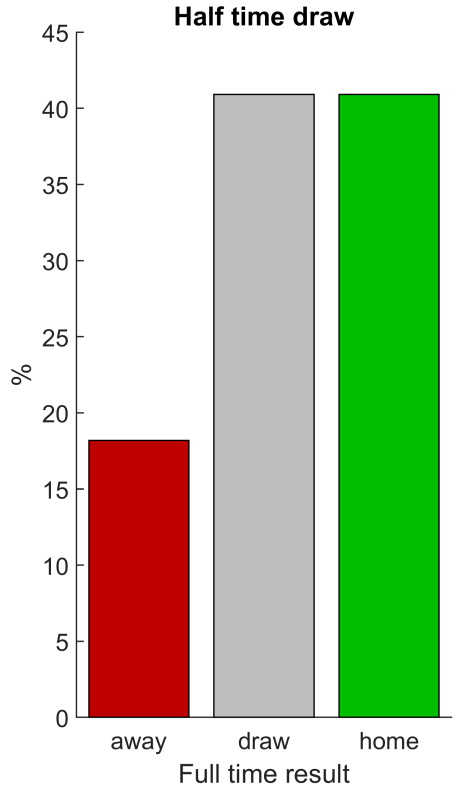
"Predict" result based on historical data

Based on historical data, we will predict the full time result depending on the half time result. We will allow to pick a home team and an away team, as well as the result after the first half. The function half2full will return a vector of ratios (in %) for away team win, draw, and home team win. On demand, it will also create some visualisation for the selection made.

```
homeTeam = 'Schalke';
awayTeam = 'Dortmund';
halfTimeWinning = 'draw';
ratioSelectedTeams = half2full(liga,homeTeam, awayTeam,halfTimeWinning,true)
```








```
ratioSelectedTeams = 1×3
   18.1818   40.9091   40.9091
```

Does this function work for each team vs. each other team? Yes, it does! We now can compare the ratios obtained for the selected teams with the ratios for all the matches with the selected half time result.

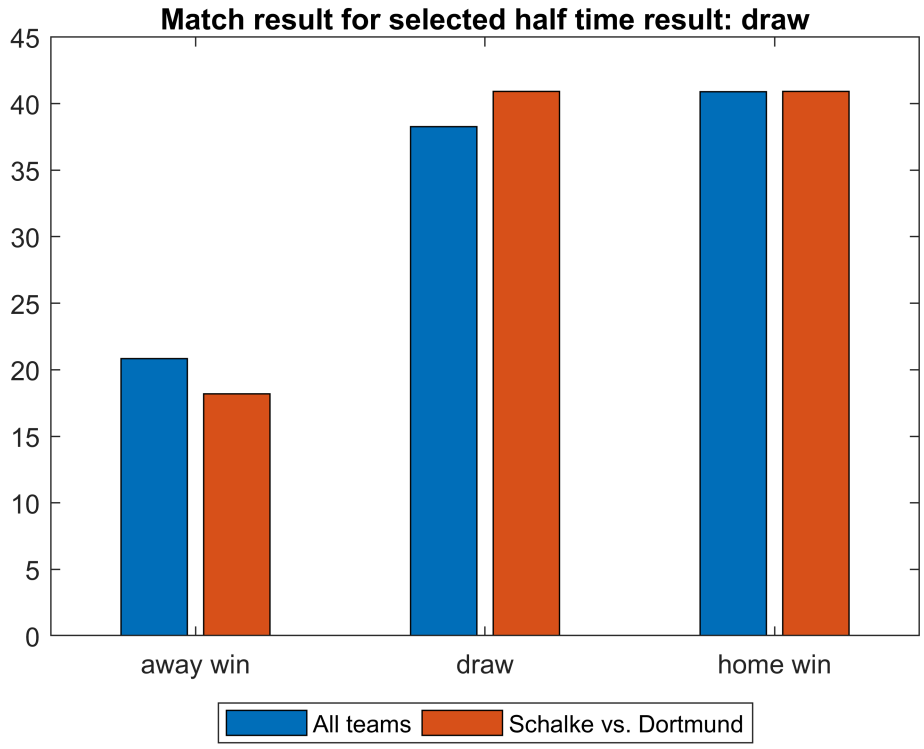
```
ratioAllTeams = half2full(liga,team,team,halfTimeWinning,false)
```

```
ratioAllTeams = 1×3
   20.8460   38.2656   40.8884
```

```
wins = categorical({'away win','draw','home win'},'Ordinal',true)
```

```
wins = 1×3 categorical array
     away win      draw      home win
```

```
figure
bar(wins,[ratioAllTeams;ratioSelectedTeams]');
legend(["All teams",[homeTeam ' vs. ' awayTeam]],...
    'Location','southoutside','Orientation','horizontal')
title(join(["Match result for selected half time result:",halfTimeWinning]))
```



half2full

The function half2full computes and visualizes the result at the final whistle for selected teams and a selected half time result.

Inputs:

`tableresults`: `table`; has to contain at least variables

- `Date` (`datetime`)
- `HomeTeam` (`categorical`)
- `AwayTeam` (`categorical`)
- `HalfTimeWinner` (`categorical`)
- `FinalWinner` (`categorical`)

`homeTeam`: `categorical`; selected team should be element in `tableresults.HomeTeam`

`awayTeam`: `categorical`; selected team should be element in `tableresults.AwayTeam`

`htWin`: `categorical`; half time result should be element in `tableresults.HalfTimeWinner`

`doplot`: `logical`; set to `true` to visualize results

Output:

`ratioResult`: `1x3 double`; contains the percentage for an away win, draw, and home win (in the specified order)

```
function ratioResult = half2full(tableresults,homeTeam,awayTeam,htWin,doplot)

% categories for comparison and visualisation
cats = categorical({'away','draw','home'},'Ordinal',true);

% Find all matches homeTeam vs. awayTeam
teamHome = ismember(tableresults.HomeTeam,homeTeam);
teamAway = ismember(tableresults.AwayTeam,awayTeam);
teamMatch = and(teamHome, teamAway);

% Find matches with the selected half time result, and the combinations with the possible
% final results
teamHomeHalfResult = teamMatch & tableresults.HalfTimeWinner == htWin;
teamHomeFinalResult = teamHomeHalfResult & tableresults.FinalWinner == cats;

% Compute the ratios for the possible results
ratioResult = sum(teamHomeFinalResult)./sum(teamHomeHalfResult) * 100;

% Visualize if asked for
if doplot
    figure

    % Historical overview of final results
    subplot(2,2,1)
    % every match between selected teams
    plot(tableresults.Date(teamMatch),zeros(nnz(teamMatch),1),'k.')
    % add circles for every match with selected result, and color it accordingly
    hold on
    colors = {[0.75 0 0],[0.75 0.75 0.75],[0 0.75 0]};
    for k = 1:3
        scatter(tableresults.Date(teamHomeFinalResult(:,k)),...
            zeros(nnz(teamHomeFinalResult(:,k)),1),...
            'filled','MarkerFaceColor',colors{k})
    end
    hold off
    xlim([min(tableresults.Date) max(tableresults.Date)])
    ylim([-0.5 1])
    legend(["Match",strcat("Full time ",string(cats))])
    title(sprintf("%s vs. %s: half time %s",homeTeam,awayTeam,string(htWin)))
    ax = gca;
    ax.XGrid = 'on';
    ax.YAxis.Visible = 'off';

    % Heatmap of goal differences at half time and full time of all matches between the
    % selected teams.
    subplot(2,2,3)
    heatmap(tableresults(teamMatch,:),'HalfGoalDiff','FinalGoalDiff',...
        'Colormap',parula,'Title','All matches');

    % Visualize ratios with colored bars
    subplot(1,2,2)
    for k = 1:3
        hold on
```

```
        bar(cats(k),ratioResult(k),'FaceColor',colors{k})
        hold off
    end
    title(join(["Half time",string(htWin)]))
    xlabel("Full time result")
    ylabel("%")
end
end
```