An Introduction to Ti*k*Z

Integrating Graphics within LaTeX

Sean Farley

LSU

October 30, 2009

LaTeX Philosophy

LaTeX is a wonderful tool for writing mathematics.

LaTeX Philosophy

LaTeX is a wonderful tool for writing mathematics. It is designed to:

- Write logically, not visually.

LaTeX Philosophy

LaTeX is a wonderful tool for writing mathematics. It is designed to:

- Write logically, not visually.
- Mark things like chapter, section, and references.

LaTeX Philosophy

LaTeX is a wonderful tool for writing mathematics. It is designed to:

- Write logically, not visually.
- Mark things like chapter, section, and references.
- Not have to worry about things like spacing, style, or numbering.

LaTeX Philosophy

LaTeX is a wonderful tool for writing mathematics. It is designed to:

- Write logically, not visually.
- Mark things like chapter, section, and references.
- Not have to worry about things like spacing, style, or numbering.

This philosophy is meant so that we don't have to worry about, for example, whether *emphasis* should be **bold**, underlined, or *italic*

LaTeX Philosophy

LaTeX is a wonderful tool for writing mathematics. It is designed to:

- Write logically, not visually.
- Mark things like chapter, section, and references.
- Not have to worry about things like spacing, style, or numbering.

This philosophy is meant so that we don't have to worry about, for example, whether *emphasis* should be **bold**, underlined, or *italic*; or, which figure number you want to reference

LaTeX Graphics

This philosophy comes to a grinding halt once you want a graphic.

$\LaTeX$ Graphics

This philosophy comes to a grinding halt once you want a graphic. How many times has this happened to you?

$\LaTeX$ Graphics

This philosophy comes to a grinding halt once you want a graphic. How many times has this happened to you?

- You had an EPS graphic but that wouldn't compile with `pdflatex`

LaTeX Graphics

This philosophy comes to a grinding halt once you want a graphic. How many times has this happened to you?

- You had an EPS graphic but that wouldn't compile with `pdflatex`
- You wanted to include an figure in the middle of text like this, ◺

LaTeX Graphics

This philosophy comes to a grinding halt once you want a graphic. How many times has this happened to you?

- You had an EPS graphic but that wouldn't compile with `pdflatex`
- You wanted to include an figure in the middle of text like this, ◺
- You had heard of `PSTricks` but thought it was too complicated, or wanted to output something other than postscript

LaTeX Graphics

This philosophy comes to a grinding halt once you want a graphic. How many times has this happened to you?

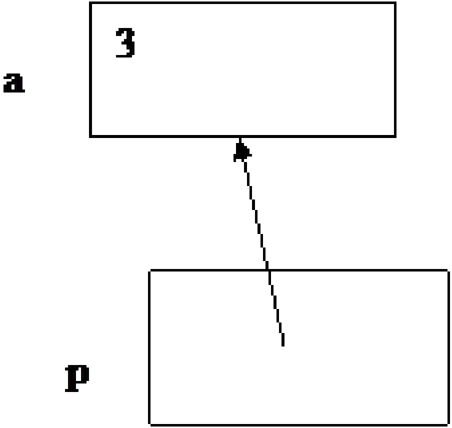
- You had an EPS graphic but that wouldn't compile with `pdflatex`
- You wanted to include an figure in the middle of text like this, ◺
- You had heard of `PSTricks` but thought it was too complicated, or wanted to output something other than postscript
- You got a plot from a collaborator but wanted to change something, for example, the tick marks, labels, etc.

Figures

Or worse, you get a graphic like this:

Figures

Or worse, you get a graphic like this:

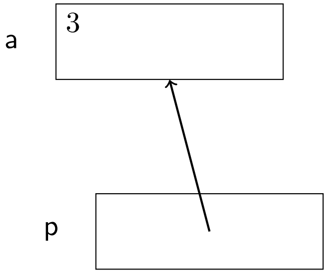


Figures

Doesn't this look much better?

Figures

Doesn't this look much better?



Enter Ti*k*Z

What is Ti*k*Z?

Enter Ti*k*Z

What is Ti*k*Z? Simply put, Ti*k*Z is a frontend for the package `PGF`, "portable graphics format."

Enter Ti*k*Z

What is Ti*k*Z? Simply put, Ti*k*Z is a frontend for the package `PGF`, "portable graphics format." In a sense, when you use `PGF` you "program" your graphics, just as you "program" your document when you use TeX.

History

Ti*k*Z is a German recursive acronym, "Ti*k*Z ist kein Zeichenprogramm," (Ti*k*Z is no drawing program).

History

Ti*k*Z is a German recursive acronym, "Ti*k*Z ist kein Zeichenprogramm," (Ti*k*Z is no drawing program). It was created by the same author of the popular package, `beamer`.

Pros and Cons

You get all the advantages of the "TEX-approach to typesetting"

- Quick creation of graphics

Pros and Cons

You get all the advantages of the "TEX-approach to typesetting"

- Quick creation of graphics
- Precise positioning

Pros and Cons

You get all the advantages of the "TEX-approach to typesetting"

- Quick creation of graphics
- Precise positioning
- Use of macros

Pros and Cons

You get all the advantages of the "TEX-approach to typesetting"

- Quick creation of graphics
- Precise positioning
- Use of macros
- Often superior typography

Pros and Cons

You get all the advantages of the "TEX-approach to typesetting"

- Quick creation of graphics
- Precise positioning
- Use of macros
- Often superior typography
- Easier collaboration

Pros and Cons

You get all the advantages of the "TEX-approach to typesetting"

- Quick creation of graphics
- Precise positioning
- Use of macros
- Often superior typography
- Easier collaboration

Disadvantages:

- Steep learning curve

Pros and Cons

You get all the advantages of the "TEX-approach to typesetting"

- Quick creation of graphics
- Precise positioning
- Use of macros
- Often superior typography
- Easier collaboration

Disadvantages:

- Steep learning curve
- No `WYSIWYG`

Ti*k*Z Environment

```
\begin{tikzpicture}
...
\end{tikzpicture}
```

TikZ Environment

```
\begin{tikzpicture}
...
\end{tikzpicture}
```

Or, to have a picture inline, use

```
\tikz ...
```

Ti*k*Z Syntax

- Every command will end with a semicolon

Ti*k*Z Syntax

- Every command will end with a semicolon
- Options will be specified between [and]

TikZ Usage

```
\path[options] <operation> ...;
```

A path is just a series of straight and curved lines

TikZ Usage

```
\path[options] <operation> ...;
```

A path is just a series of straight and curved lines, but it is not yet specified what should happen with it.

TikZ Usage

One can draw a path

```
\path[draw] <operation> ...;
```

TikZ Usage

One can draw a path, fill a path

```
\path[fill] <operation> ...;
```

TikZ Usage

```
\path[shade] <operation> ...;
```

One can draw a path, fill a path, shade it

TikZ Usage

```
\path[clip] <operation> ...;
```

One can draw a path, fill a path, shade it, clip it

TikZ Usage

```
\path[draw,shade] <operation>
...;
```

One can draw a path, fill a path, shade it, clip it, or do any combination of these.

Ti*k*Z Shortcuts

Ti*k*Zalso has shortcuts for `draw`, `fill`, `shade`, etc.

Ti*k*Z Shortcuts

Ti*k*Zalso has shortcuts for `draw`, `fill`, `shade`, etc.

- `\path[draw]` = `\draw`

TikZ Shortcuts

TikZalso has shortcuts for `draw`, `fill`, `shade`, etc.

- `\path[draw] = \draw`
- `\path[fill] = \fill`

TikZ Shortcuts

TikZalso has shortcuts for `draw`, `fill`, `shade`, etc.

- `\path[draw]` = `\draw`
- `\path[fill]` = `\fill`
- `\path[shade]` = `\shade`

TikZ Operations

Operations in TikZ include:

- `coordinate`

TikZ Operations

Operations in TikZ include:

- `coordinate`
- `--, -|, |-`

Ti*k*Z Operations

Operations in Ti*k*Z include:

- `coordinate`
- `--`, `-|`, `|-`
- `rectangle`

TikZ Operations

Operations in TikZ include:

- `coordinate`
- `--`, `-|`, `|-`
- `rectangle`
- `circle`

TikZ Operations

Operations in TikZ include:

- `coordinate`
- `--`, `-|`, `|-`
- `rectangle`
- `circle`
- `etc.`

Drawing Paths with Coordinates

Let's begin with drawing a line.

Drawing Paths with Coordinates

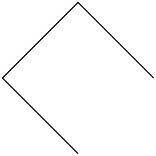
Let's begin with drawing a line.

```
\draw (1,0) -- (0,1) -- (-1,0) -- (0,1);
```

Drawing Paths with Coordinates

Let's begin with drawing a line.

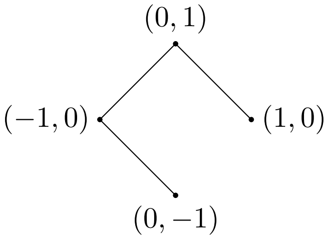
```
\draw (1,0) -- (0,1) -- (-1,0) -- (0,1);
```



Drawing Paths with Coordinates

Let's begin with drawing a line.

```
\draw (1,0) -- (0,1) -- (-1,0) -- (0,1);
```



Drawing Paths with Coordinates

To complete a cycle, use the keyword `cycle`

Drawing Paths with Coordinates

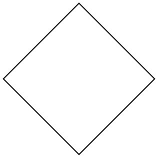
To complete a cycle, use the keyword `cycle`

```
\draw (1,0) -- (0,1) -- (-1,0) -- (0,1) -- cycle;
```

Drawing Paths with Coordinates

To complete a cycle, use the keyword `cycle`

```
\draw (1,0) -- (0,1) -- (-1,0) -- (0,1) -- cycle;
```



Using Polar Coordinates

To use polar coordinates, specify the radius and angle in degrees, as `(degrees:radius)`

Using Polar Coordinates

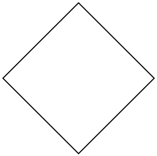
To use polar coordinates, specify the radius and angle in degrees, as `(degrees:radius)`

```
\draw (0:1) -- (90:1) -- (180:1) -- (270:1) -- cycle;
```

Using Polar Coordinates

To use polar coordinates, specify the radius and angle in degrees, as `(degrees:radius)`

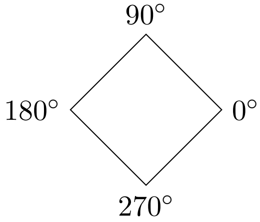
```
\draw (0:1) -- (90:1) -- (180:1) -- (270:1) -- cycle;
```



Using Polar Coordinates

To use polar coordinates, specify the radius and angle in degrees, as `(degrees:radius)`

```
\draw (0:1) -- (90:1) -- (180:1) -- (270:1) -- cycle;
```



Arcs

Let's use a built-in line shape called arc

Arcs

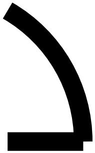
Let's use a built-in line shape called arc

```
\draw (0:1cm) -- (0:2cm)
```

Arcs

Let's use a built-in line shape called arc

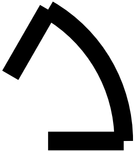
```
\draw (0:1cm) -- (0:2cm)
arc (0:60:2cm)
```



Arcs

Let's use a built-in line shape called arc

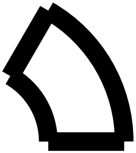
```
\draw (0:1cm) -- (0:2cm)
arc (0:60:2cm) -- (60:1cm)
```



Arcs

Let's use a built-in line shape called arc

```
\draw (0:1cm) -- (0:2cm)
arc (0:60:2cm) -- (60:1cm)
arc (60:0:1cm) -- cycle;
```



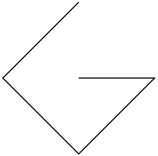
Relative Offset

We may also reference points by a relative offset.

Relative Offset

We may also reference points by a relative offset.

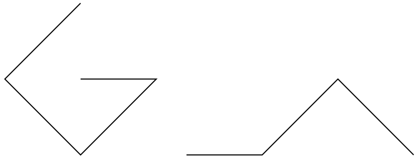
```
\draw (0,0) -- +(1,0) -- +(0,-1) -- +(-1,0) -- +(0,1);
```



Relative Offset

We may also reference points by a relative offset.

```
\draw (0,0) -- +(1,0) -- +(0,-1) -- +(-1,0) -- +(0,1);
\draw (0,0) -- ++(1,0) -- ++(1,1) -- ++(1,-1);
```



Built-in Shapes

There are basic shapes that Ti*k*Z provides

Built-in Shapes

There are basic shapes that Ti*k*Z provides

- Rectangles

Built-in Shapes

There are basic shapes that Ti*k*Z provides

- Rectangles
- Grids

Built-in Shapes

There are basic shapes that Ti*k*Z provides

- Rectangles
- Grids
- Circles and ellipses

Built-in Shapes

There are basic shapes that Ti*k*Z provides

- Rectangles
- Grids
- Circles and ellipses
- Arcs

Built-in Shapes

There are basic shapes that Ti*k*Z provides

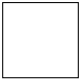
- Rectangles
- Grids
- Circles and ellipses
- Arcs
- Bézier curves

Rectangle Example

```
\draw (0,0) rectangle (1,1);
```

Rectangle Example

```
\draw (0,0) rectangle (1,1);
```



Rectangle Example

```
\draw (0,0) rectangle (1,1) rectangle (4,2);
```

Rectangle Example

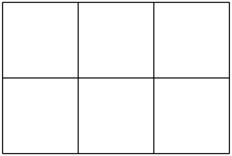
```
\draw (0,0) rectangle (1,1) rectangle (4,2);
```

Grid Example

```
\draw (0,0) grid (3,2);
```

Grid Example

```
\draw (0,0) grid (3,2);
```

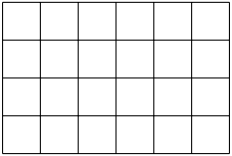


Grid Example

```
\draw[step=.5cm] (0,0) grid (3,2);
```

Grid Example

```
\draw[step=.5cm] (0,0) grid (3,2);
```

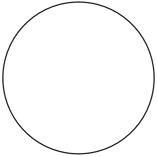


Circle Example

```
\draw (0,0) circle (1cm);
```

Circle Example

```
\draw (0,0) circle (1cm);
```

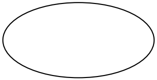


Ellipse Example

```
\draw (0,0) ellipse [x radius=1cm,y radius=.5cm];
```

Ellipse Example

```
\draw (0,0) ellipse [x radius=1cm,y radius=.5cm];
```



Arc Example

```
\draw (0,0) arc (0:30:3cm);
```

Arc Example

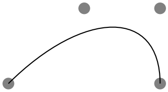
```
\draw (0,0) arc (0:30:3cm);
```

Bézier Curves Example

```
\draw (0,0) .. controls (1,1) and (2,1) .. (2,0);
```

Bézier Curves Example

```
\draw (0,0) .. controls (1,1) and (2,1) .. (2,0);
```



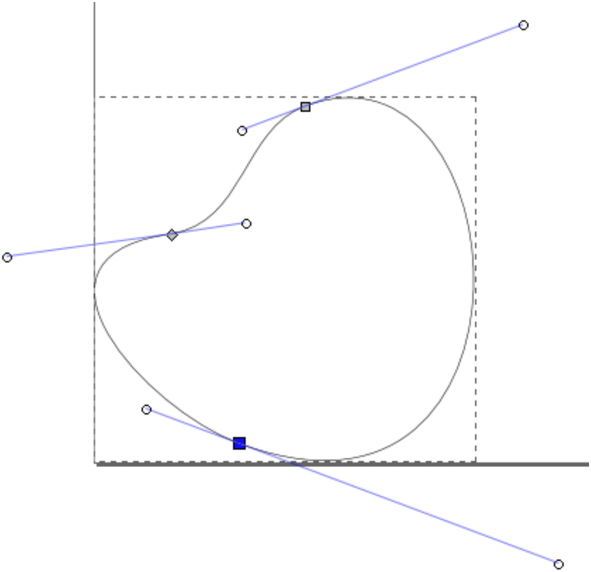
Inkscape

`Inkscape` is a free and open-source program for vector drawing.
Available on Windows/Linux/Mac.

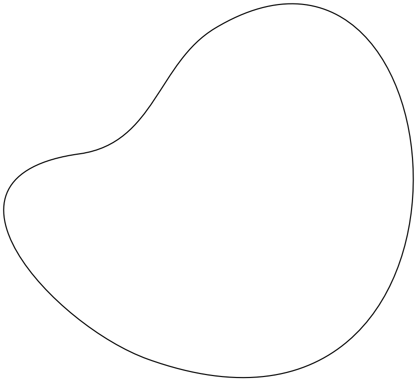
Inkscape

`Inkscape` is a free and open-source program for vector drawing. Available on Windows/Linux/Mac. With `Inkscape` you can draw a Bézier curve and export it to Ti*k*Z.

Inkscape Example



Inkscape Example



Inkscape Example

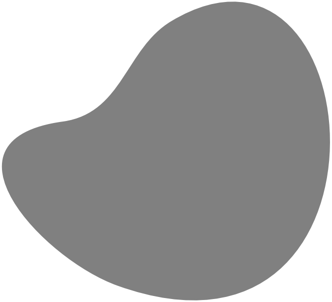


Figure: A 2-D domain, Ω

Coordinate Labeling

```
\coordinate (P1) at (0,0);
\coordinate (P2) at (1,1);
```

Coordinate Labeling

```
\coordinate (P1) at (0,0);
\coordinate (P2) at (1,1);
```

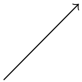
This draws nothing to see, but instead just places a named reference to the points

Coordinate Referencing

```
\draw[->] (P1) -- (P2);
```

Coordinate Referencing

```
\draw[->] (P1) -- (P2);
```

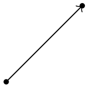


Coordinate Referencing

```
\fill (P1) circle (1pt);
\fill (P2) circle (1pt);
```

Coordinate Referencing

```
\fill (P1) circle (1pt);
\fill (P2) circle (1pt);
```



Nodes

Using `\node` allows text to placed in a Ti*k*Z picture.

Nodes

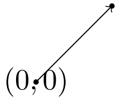
Using `\node` allows text to placed in a Ti*k*Z picture.

`\node at (P1) {`$(0,0)$`};`

Nodes

Using `\node` allows text to placed in a Ti*k*Z picture.

`\node at (P1) {`$(0,0)$`};`



Nodes

What about text positioning?

Nodes

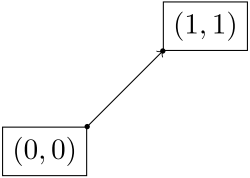
What about text positioning?

`\node[below left,draw] at (P1) {`$(0,0)$`};`

Nodes

What about text positioning?

`\node[below left,draw] at (P1) {`$(0,0)$`};`



Text Nodes and Styles

```
every node/.style={draw,align=center,rounded corners}
```

Text Nodes and Styles

```
every node/.style={draw,align=center,rounded corners}
uedge/.style={fill=blue!20}
```

Text Nodes and Styles

```
every node/.style={draw,align=center,rounded corners}
uedge/.style={fill=blue!20}
\node[text width=10cm,uedge] {UEDGE Driver ...};
```

Text Nodes and Styles

```
every node/.style={draw,align=center,rounded corners}
uedge/.style={fill=blue!20}
\node[text width=10cm,uedge] {UEDGE Driver ...};
```

UEDGE Driver + Timestepping + Parallel Partitioning

Text Nodes and Styles

```
every node/.style={draw,align=center,rounded corners}
uedge/.style={fill=blue!20}
\node[text width=10cm,uedge] {UEDGE Driver ...};
```

UEDGE Driver + Timestepping + Parallel Partitioning

```
petsc/.style=fill=yellow!60,below=.5cm
```

Text Nodes and Styles

```
every node/.style={draw,align=center,rounded corners}
uedge/.style={fill=blue!20}
\node[text width=10cm,uedge] {UEDGE Driver ...};
```

UEDGE Driver + Timestepping + Parallel Partitioning

```
petsc/.style=fill=yellow!60,below=.5cm
\node[text width=6cm,petsc] Nonlinear Solvers (SNES);
```

Text Nodes and Styles

```
every node/.style={draw,align=center,rounded corners}
uedge/.style={fill=blue!20}
\node[text width=10cm,uedge] {UEDGE Driver ...};
```

UEDGE Driver + Timestepping + Parallel Partitioning

```
petsc/.style=fill=yellow!60,below=.5cm
\node[text width=6cm,petsc] Nonlinear Solvers (SNES);
```

Nonlinear Solvers (SNES)

Arrow Options

What about different types of arrows?

Arrow Options

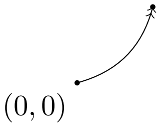
What about different types of arrows?

```
\draw[->>] (P1) to[bend right] (P2);
```

Arrow Options

What about different types of arrows?

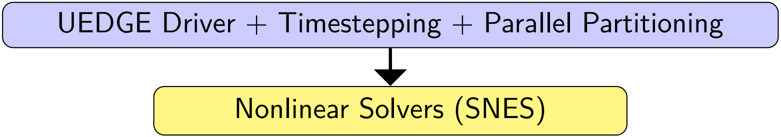
```
\draw[-»] (P1) to[bend right] (P2);
```



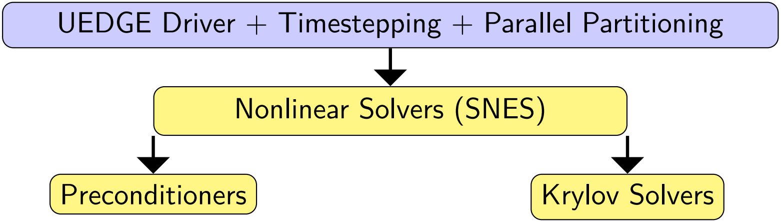
UEDGE

UEDGE Driver + Timestepping + Parallel Partitioning

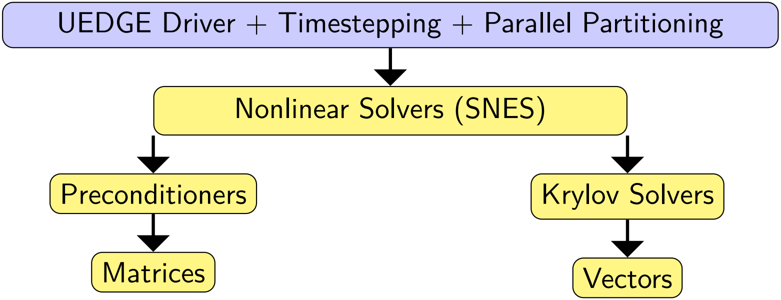
UEDGE


UEDGE Driver + Timestepping + Parallel Partitioning
Nonlinear Solvers (SNES)

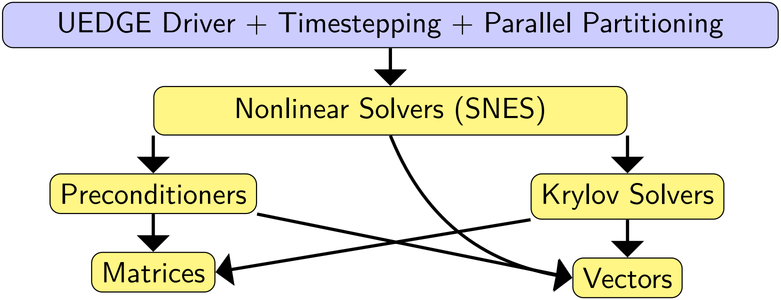
UEDGE


UEDGE Driver + Timestepping + Parallel Partitioning
Nonlinear Solvers (SNES)
Preconditioners
Krylov Solvers

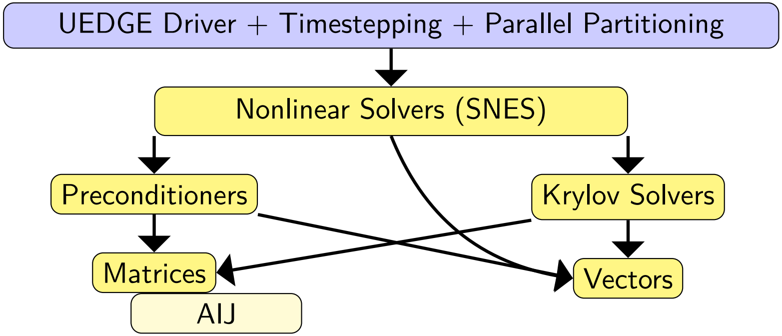
UEDGE


UEDGE Driver + Timestepping + Parallel Partitioning
Nonlinear Solvers (SNES)
Preconditioners
Krylov Solvers
Matrices
Vectors

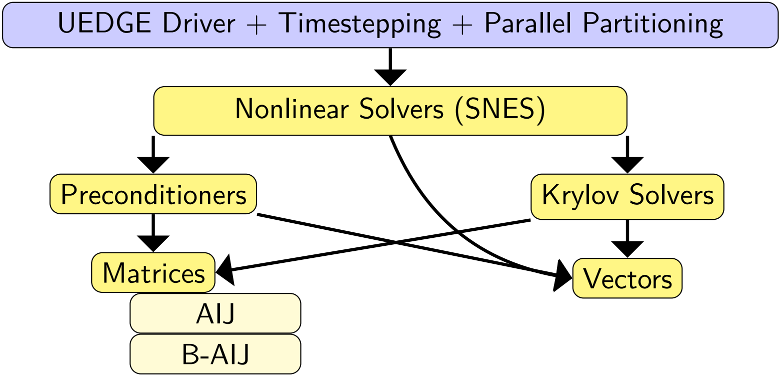
UEDGE


UEDGE Driver + Timestepping + Parallel Partitioning
Nonlinear Solvers (SNES)
Preconditioners
Krylov Solvers
Matrices
Vectors

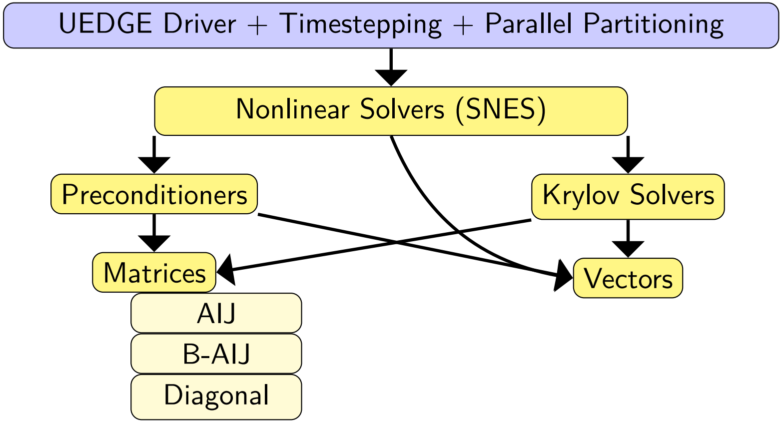
UEDGE


UEDGE Driver + Timestepping + Parallel Partitioning
Nonlinear Solvers (SNES)
Preconditioners
Krylov Solvers
Matrices
AIJ
Vectors

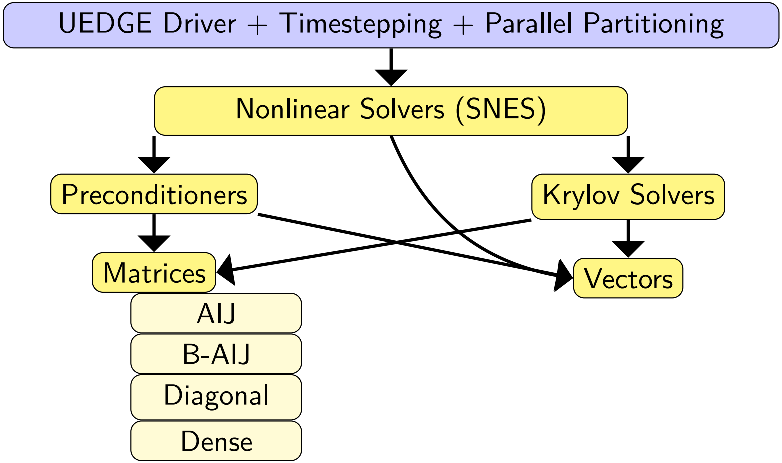
UEDGE


UEDGE Driver + Timestepping + Parallel Partitioning
Nonlinear Solvers (SNES)
Preconditioners
Krylov Solvers
Matrices
Vectors
AIJ
B-AIJ

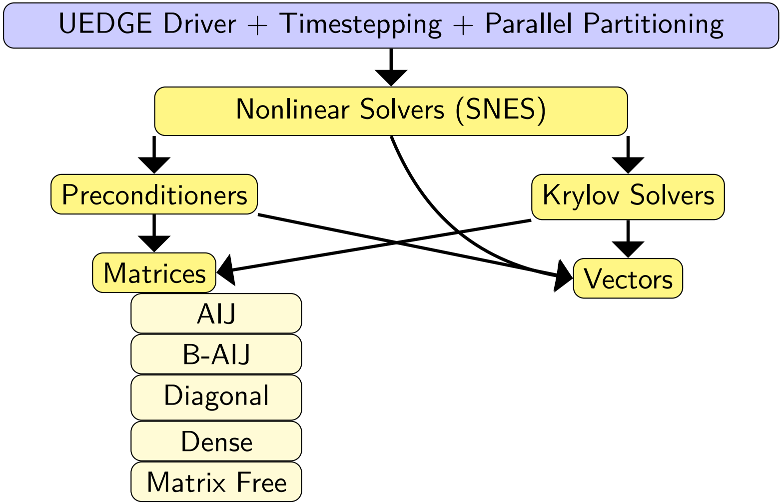
UEDGE


UEDGE Driver + Timestepping + Parallel Partitioning
Nonlinear Solvers (SNES)
Preconditioners
Krylov Solvers
Matrices
Vectors
AIJ
B-AIJ
Diagonal

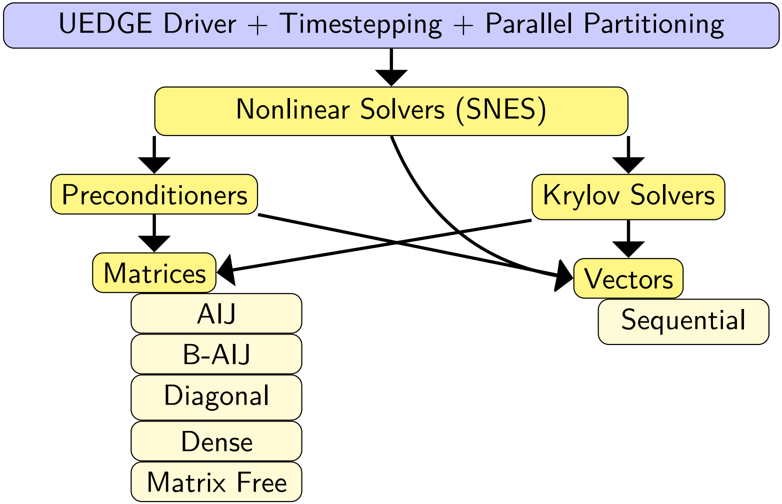
UEDGE


UEDGE Driver + Timestepping + Parallel Partitioning
Nonlinear Solvers (SNES)
Preconditioners
Krylov Solvers
Matrices
Vectors
AIJ
B-AIJ
Diagonal
Dense

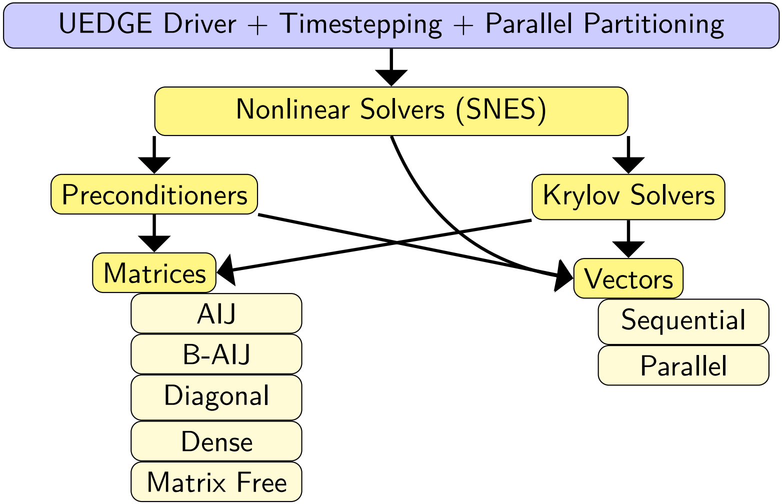
UEDGE


UEDGE Driver + Timestepping + Parallel Partitioning
Nonlinear Solvers (SNES)
Preconditioners
Krylov Solvers
Matrices
Vectors
AIJ
B-AIJ
Diagonal
Dense
Matrix Free

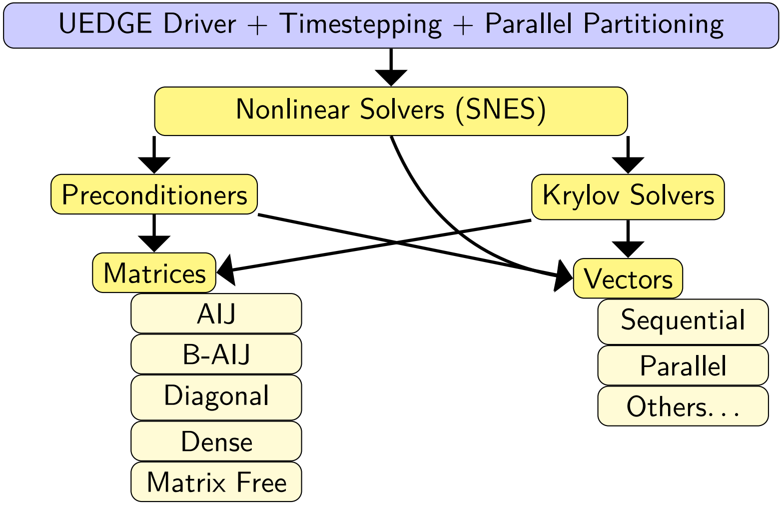
UEDGE


UEDGE Driver + Timestepping + Parallel Partitioning
Nonlinear Solvers (SNES)
Preconditioners
Krylov Solvers
Matrices
Vectors
AIJ
B-AIJ
Diagonal
Dense
Matrix Free
Sequential

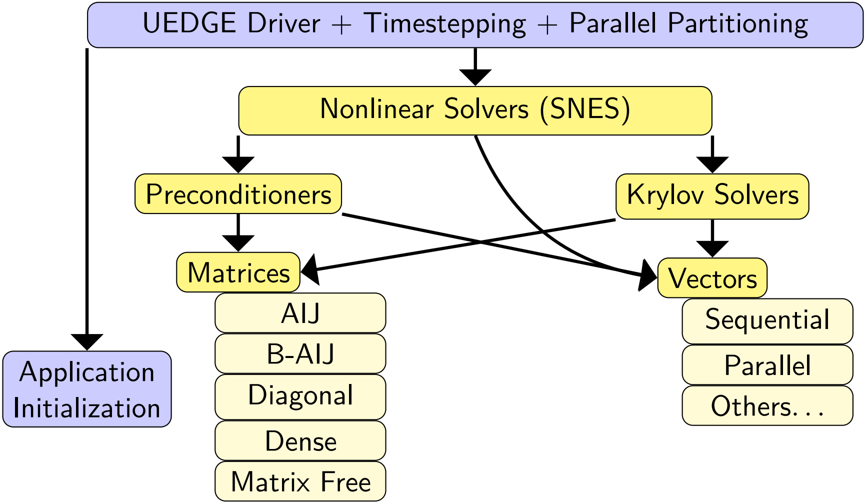
UEDGE


UEDGE Driver + Timestepping + Parallel Partitioning
Nonlinear Solvers (SNES)
Preconditioners
Krylov Solvers
Matrices
Vectors
AIJ
B-AIJ
Diagonal
Dense
Matrix Free
Sequential
Parallel

UEDGE


UEDGE Driver + Timestepping + Parallel Partitioning
Nonlinear Solvers (SNES)
Preconditioners
Krylov Solvers
Matrices
AIJ
B-AIJ
Diagonal
Dense
Matrix Free
Vectors
Sequential
Parallel
Others...

UEDGE

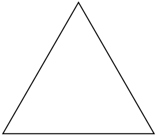

UEDGE Driver + Timestepping + Parallel Partitioning
Nonlinear Solvers (SNES)
Preconditioners
Krylov Solvers
Matrices
AIJ
B-AIJ
Diagonal
Dense
Matrix Free
Vectors
Sequential
Parallel
Others...
Application Initialization

The Lagrange Element

Let's now draw something useful.

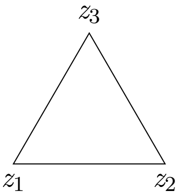
The Lagrange Element

```
\coordinate (z1) at (0,0);
\coordinate (z2) at (2,0);
\coordinate (z3) at (1,1.73);
```



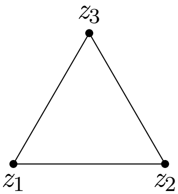
The Lagrange Element

```
\node[below] at (z1) {$z_1$};
\node[below] at (z2) {$z_2$};
\node[above] at (z3) {$z_3$};
```



The Lagrange Element

```
\fill (z1) circle (1.5pt);
\fill (z2) circle (1.5pt);
\fill (z3) circle (1.5pt);
```

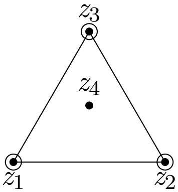


The Cubic Hermite Element

How about the cubic Hermite element?

The Cubic Hermite Element

```
\draw (z1) circle (3pt);
\draw (z2) circle (3pt);
\draw (z3) circle (3pt);
```



Commutative Diagram Example

Label and reference point in one step,

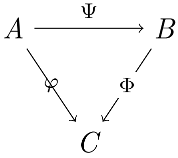
Commutative Diagram Example

```
\node at (0,0) (A) {$A$};
\node at (2,0) (B) {$B$};
\node at (1,-1.5) (C) {$C$};
```

$$\begin{tikzcd}
A & & B \\
& C &
\end{tikzcd}$$

Commutative Diagram Example

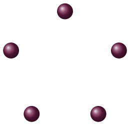
```
\path[->,font=\scriptsize]
(A) edge node {$\varphi$}(C)
(A) edge node[auto] {$\Psi$}(B)
(B) edge node[fill=white,inner sep=2pt] {$\Phi$}(C);
```



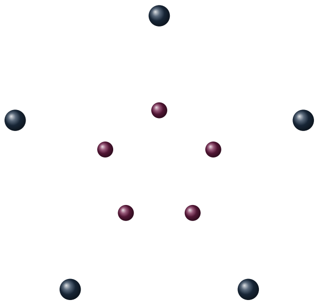
LSU Math Logo

We now are able to draw the LSU Math Logo

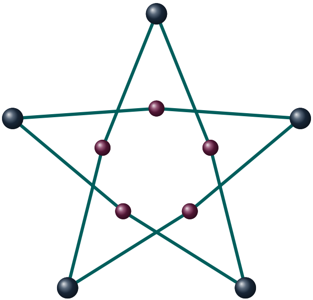
LSU Math Logo



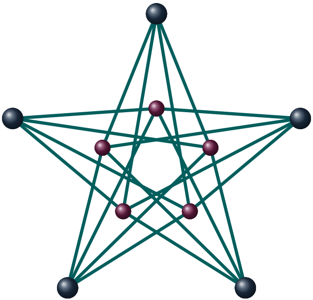
LSU Math Logo



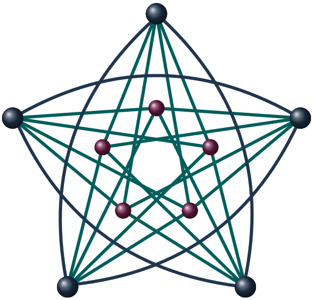
LSU Math Logo



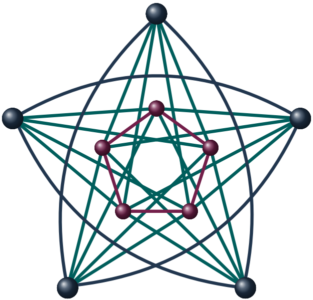
LSU Math Logo



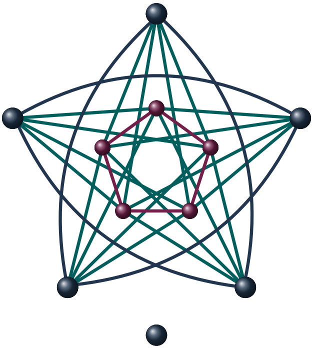
LSU Math Logo



LSU Math Logo



LSU Math Logo

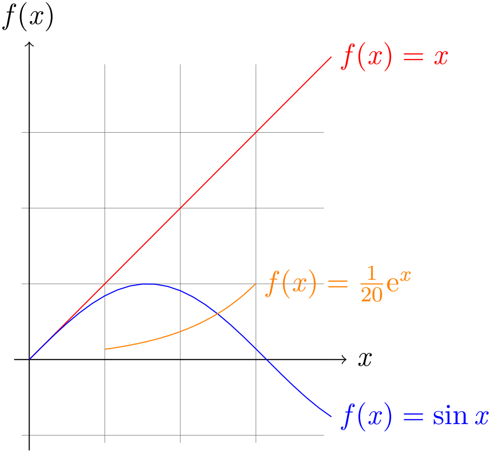


Plotting

What about plotting?

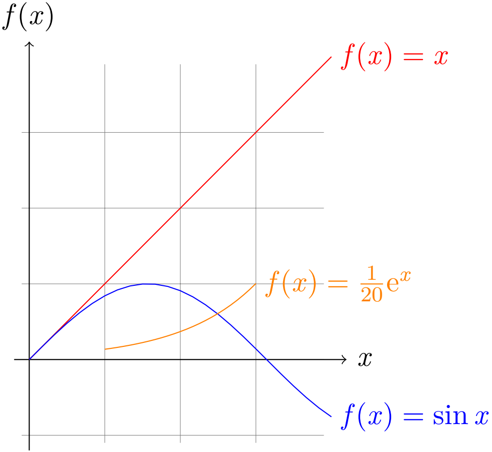
Plotting

```
\draw[color=blue]
plot (\x,{sin(\x r)}) node[right] {$f(x) = \sin x$};
```



Plotting

```
\draw[color=orange,domain=1:3]
plot (\x,{0.05*exp(\x)}) node[right] {$f(x) = \frac{1}{20}e^x$};
```



Parametric Plotting

We may also generate a parametric plot

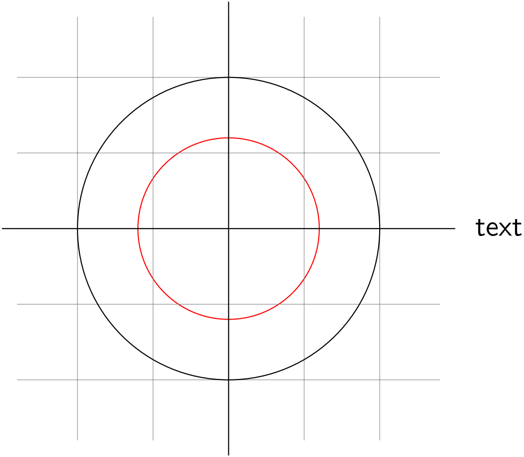
Parametric Plotting

```
\draw[scale=0.5,domain=-1.3:1.3,smooth,variable=\t]
plot ({\t^5+sin(2*pi*\t r)},{\t + exp(\t)});
```

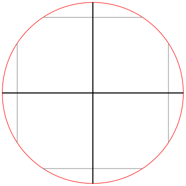
Data Plotting

```
\draw[scale=.5] plot[mark=x,only marks] file {sin.table};
```

Clipping Example

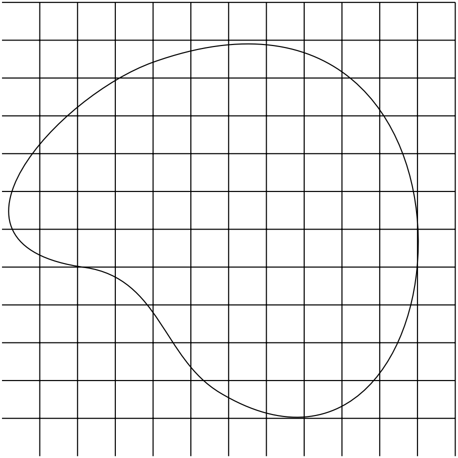


Clipping Example

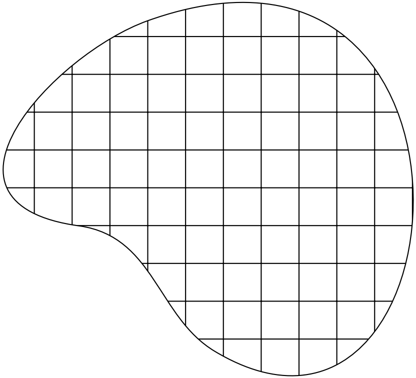


text

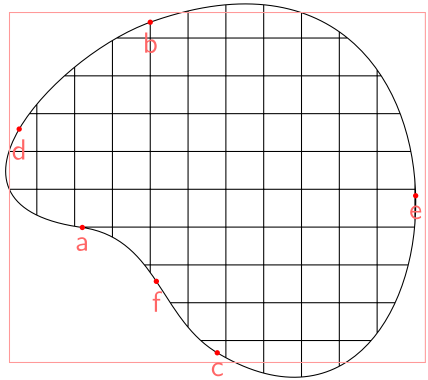
Scope



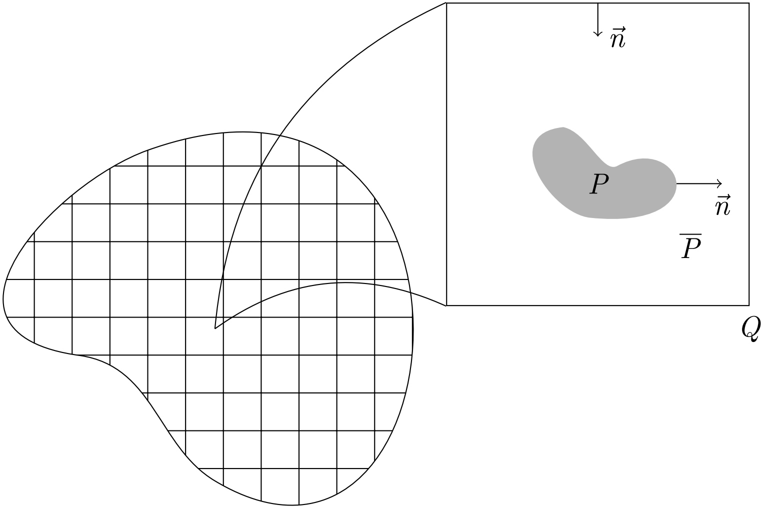
Scope



Scope



Scope



Interactive Beamer Arrows

$$\vec{a}_p = \vec{a}_o + \frac{{}^b d^2}{dt^2}\vec{r} + 2\vec{\omega}_{ib} \times \frac{{}^b d}{dt}\vec{r} + \vec{\alpha}_{ib} \times \vec{r} + \vec{\omega}_{ib} \times (\vec{\omega}_{ib} \times \vec{r}) \tag{1}$$

Interactive Beamer Arrows

- Coriolis acceleration

$$\vec{a}_p = \vec{a}_o + \frac{{}^b d^2}{dt^2}\vec{r} + \boxed{2\vec{\omega}_{ib} \times \frac{{}^b d}{dt}\vec{r}} + \quad \vec{\alpha}_{ib} \times \vec{r} \quad + \; \vec{\omega}_{ib} \times (\vec{\omega}_{ib} \times \vec{r}) \tag{1}$$

Interactive Beamer Arrows

- Coriolis acceleration

$$\vec{a}_p = \vec{a}_o + \frac{{}^b d^2}{dt^2}\vec{r} + \boxed{2\vec{\omega}_{ib} \times \frac{{}^b d}{dt}\vec{r}} + \vec{\alpha}_{ib} \times \vec{r} + \vec{\omega}_{ib} \times (\vec{\omega}_{ib} \times \vec{r}) \qquad (1)$$

- Transversal acceleration

Interactive Beamer Arrows

- Coriolis acceleration

$$\vec{a}_p = \vec{a}_o + \frac{{}^b d^2}{dt^2}\vec{r} + \boxed{2\vec{\omega}_{ib} \times \frac{{}^b d}{dt}\vec{r}} + \vec{\alpha}_{ib} \times \vec{r} + \boxed{\vec{\omega}_{ib} \times (\vec{\omega}_{ib} \times \vec{r})} \qquad (1)$$

- Transversal acceleration
- Centripetal acceleration

References

- PGF Manual
- texamples.net: A very good site with TEX+TikZ Examples
- Link for Commutative Diagrams