L26: Taylor Series

And introduction to the final programming project

Lucas A. J. Bastien

E7 Spring 2017, University of California at Berkeley

March 22, 2017

Version: release

Announcements

Lab 09 is due on March 24 at 12 pm (noon)

Today:

- One more thing about interpolation (see updated slides of L25)
- Taylor series (chapter 15)
- Introduction to the final programming project

Friday:

- Pseudo-random numbers
- Discussion

Announcements

Lab 09 is due on March 24 at 12 pm (noon)

Today:

- One more thing about interpolation (see updated slides of L25)
- Taylor series (chapter 15)
- Introduction to the final programming project

Friday:

- Pseudo-random numbers
- Discussion

Next Week: Spring break!

- Get some rest, have some fun, see friends and family
- Get a head start on lab 10 and on your E7 project

Announcements

Lab 09 is due on March 24 at 12 pm (noon)

Today:

- One more thing about interpolation (see updated slides of L25)
- Taylor series (chapter 15)
- Introduction to the final programming project

Friday:

- Pseudo-random numbers
- Discussion

Next Week: Spring break!

- Get some rest, have some fun, see friends and family
- Get a head start on lab 10 and on your E7 project

After Spring break:

- Numerical differentiation (Chapter 17)
- Numerical integration (Chapter 18)

Taylor series

Consider a real-valued function that is C^∞ (*i.e.* infinitely differentiable) over some interval I. Pick a point $a \in I$. Then, for any value x of this interval I:

$$\begin{aligned} f(x) &= f(a) + \frac{f'(a)}{1!}(x-a) + \frac{f''(a)}{2!}(x-a)^2 + \frac{f'''(a)}{3!}(x-a)^3 + \dots \\ &= f(a) + \sum_{k=1}^{\infty} \frac{f^{(k)}(a)}{k!}(x-a)^k \end{aligned}$$

$f^{(k)}$ is the k^{th} derivative of f

Taylor series

Consider a real-valued function that is C^∞ (*i.e.* infinitely differentiable) over some interval I. Pick a point $a \in I$. Then, for any value x of this interval I:

$$\begin{aligned} f(x) &= f(a) + \frac{f'(a)}{1!}(x-a) + \frac{f''(a)}{2!}(x-a)^2 + \frac{f'''(a)}{3!}(x-a)^3 + \dots \\ &= f(a) + \sum_{k=1}^{\infty} \frac{f^{(k)}(a)}{k!}(x-a)^k \end{aligned}$$

$f^{(k)}$ is the k^{th} derivative of f

The equation above is the **Taylor series of f centered at a**. **It gives an expression for $f(x)$ as a function of x and of the values of the function and its derivatives at a**

Truncated Taylor series

$$f(x) = f(a) + \sum_{k=1}^{\infty} \frac{f^{(k)}(a)}{k!}(x-a)^k$$
$$\approx f(a) + \sum_{k=1}^{n} \frac{f^{(k)}(a)}{k!}(x-a)^k$$

One often uses truncated Taylor series "with m terms" (i.e. the constant term plus the first $m-1$ terms of the infinite sum above) to approximate functions

Truncated Taylor series

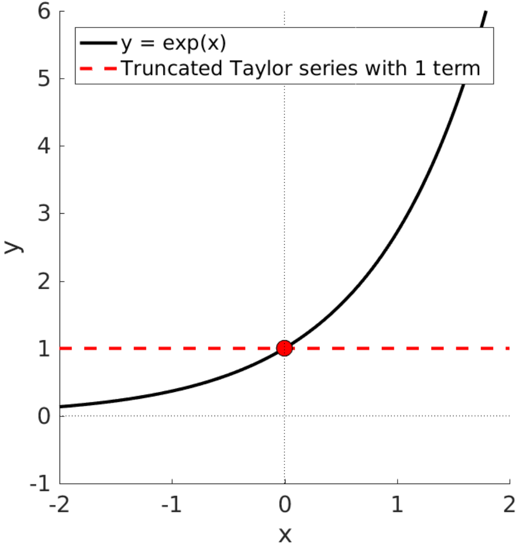
$$f(x) = f(a) + \sum_{k=1}^{\infty} \frac{f^{(k)}(a)}{k!}(x-a)^k$$
$$\approx f(a) + \sum_{k=1}^{n} \frac{f^{(k)}(a)}{k!}(x-a)^k$$

One often uses truncated Taylor series "with m terms" (i.e. the constant term plus the first $m-1$ terms of the infinite sum above) to approximate functions

In general:

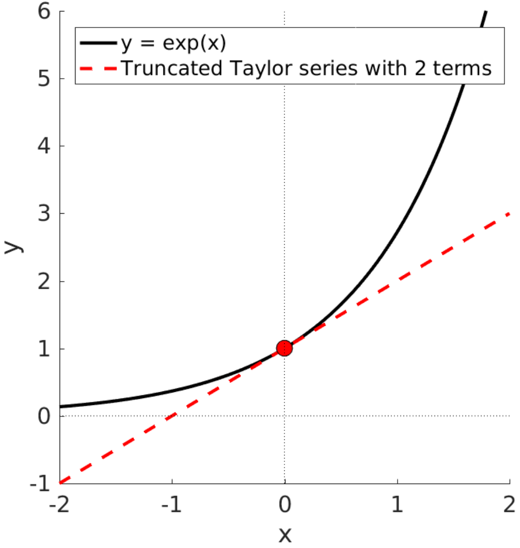
- The closer x is to the center point a, the smaller the error
- When using more terms of the series (*i.e.* higher n), the above approximation holds for a larger range of x away from the center point a (see figures in following slides)

Function: $x \mapsto \exp(x)$, **center point** $a = 0$



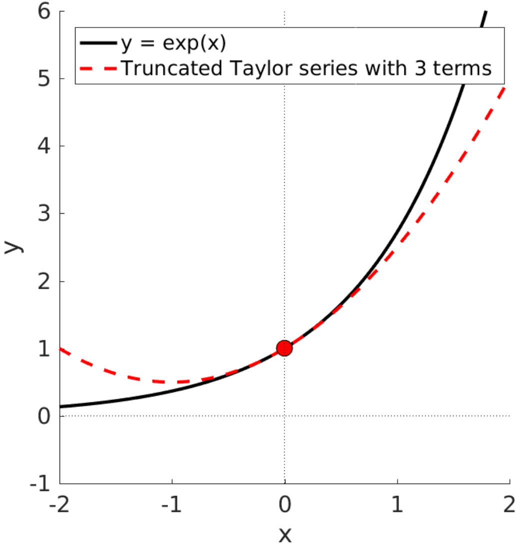
Truncated Taylor series: examples

Function: $x \mapsto \exp(x)$, **center point** $a = 0$



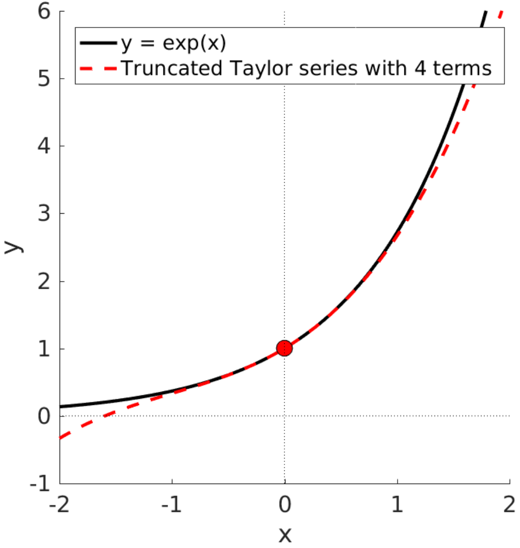
Truncated Taylor series: examples

Function: $x \mapsto \exp(x)$**, center point** $a = 0$



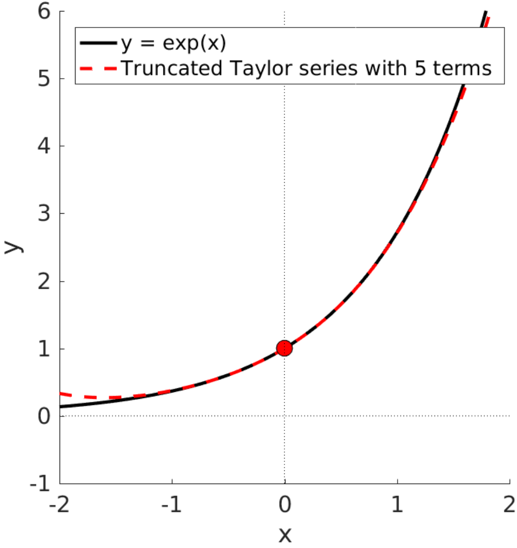
Truncated Taylor series: examples

Function: $x \mapsto \exp(x)$**, center point** $a = 0$



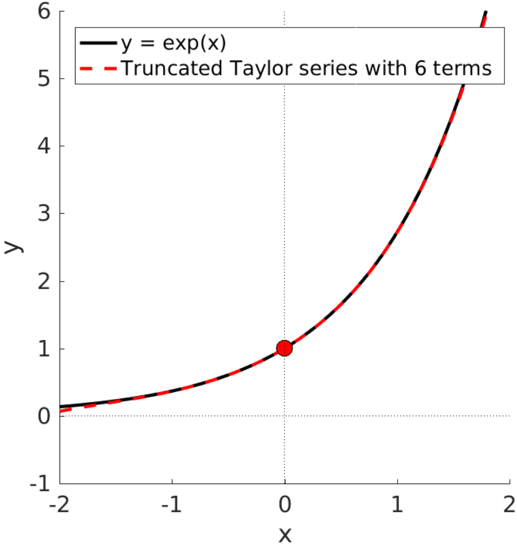
Truncated Taylor series: examples

Function: $x \mapsto \exp(x)$**, center point** $a = 0$



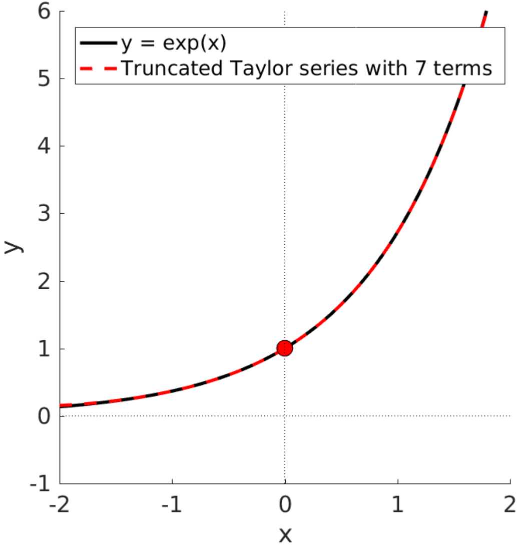
Truncated Taylor series: examples

Function: $x \mapsto \exp(x)$**, center point** $a = 0$



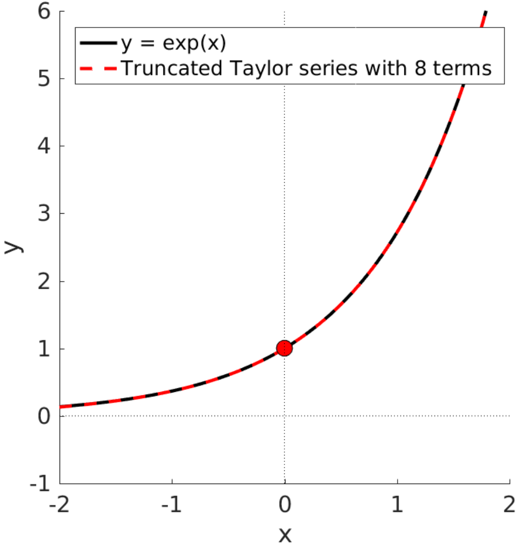
Truncated Taylor series: examples

Function: $x \mapsto \exp(x)$**, center point** $a = 0$



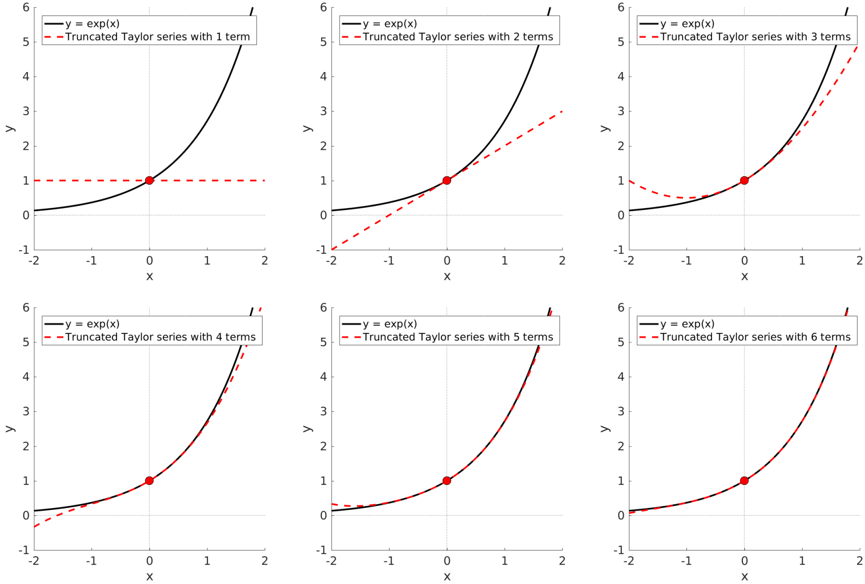
Truncated Taylor series: examples

Function: $x \mapsto \exp(x)$, **center point** $a = 0$



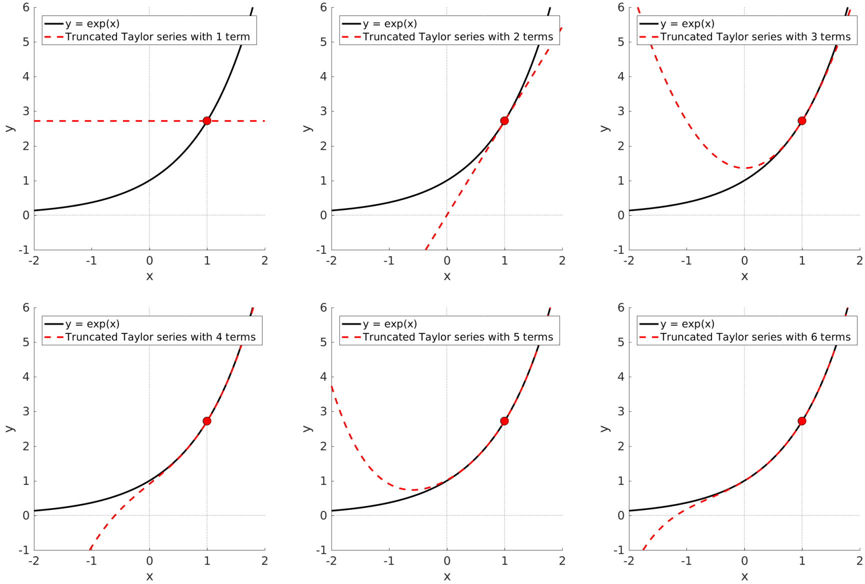
Truncated Taylor series: examples

Function: $x \mapsto \exp(x)$, **center point** $a = 0$



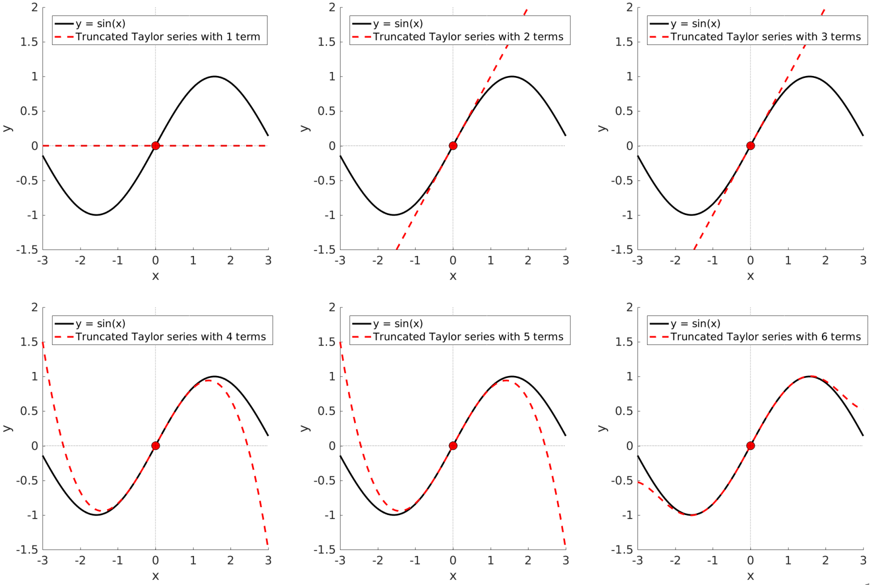
Truncated Taylor series: examples

Function: $x \mapsto \exp(x)$, center point $a = 1$



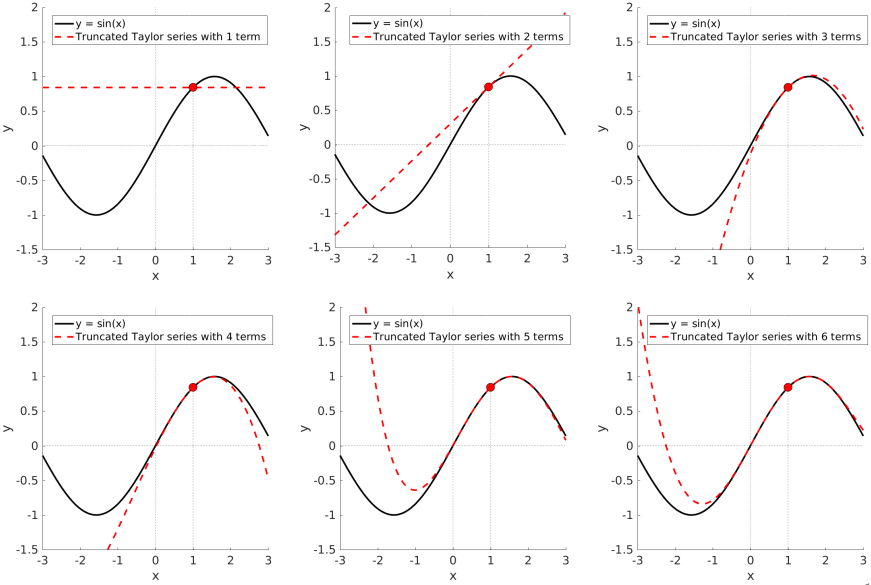
Truncated Taylor series: examples

Function: $x \mapsto \sin(x)$, **center point** $a = 0$



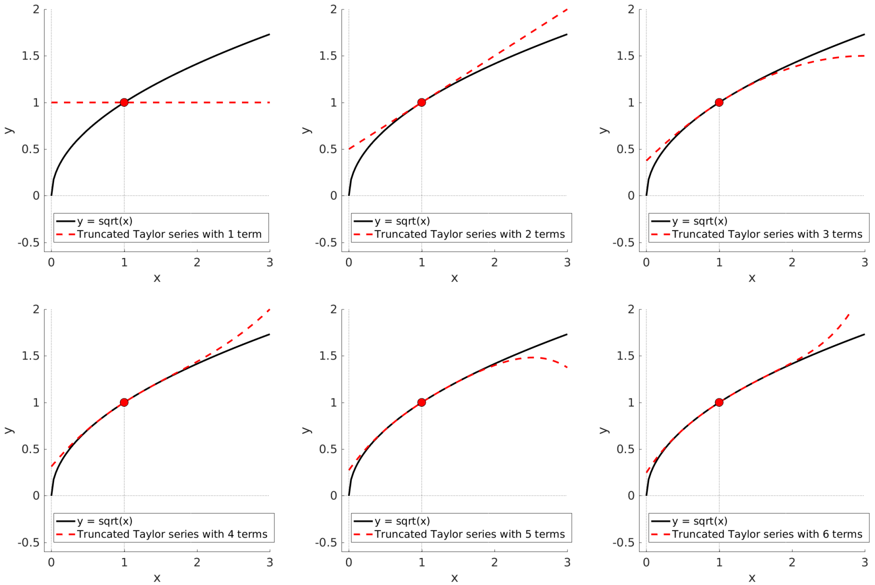
Truncated Taylor series: examples

Function: $x \mapsto \sin(x)$, center point $a = 1$



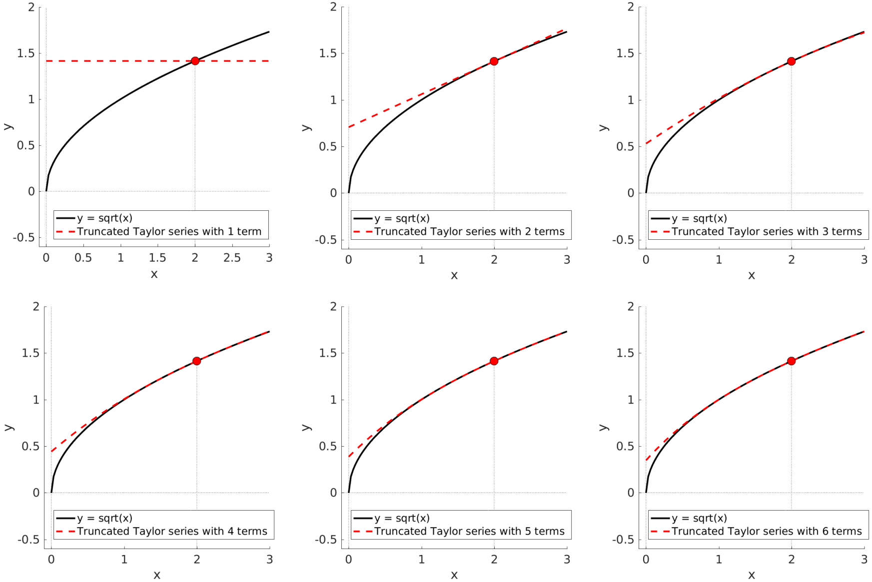
Truncated Taylor series: examples

Function: $x \mapsto \sqrt{x}$, center point $a = 1$

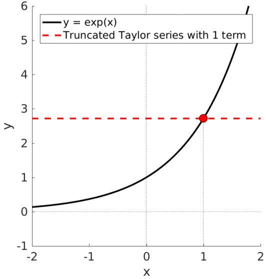


Truncated Taylor series: examples

Function: $x \mapsto \sqrt{x}$**, center point** $a = 2$



Example with function: $x \mapsto \exp(x)$ **and center point** $a = 1$

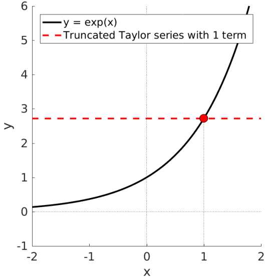


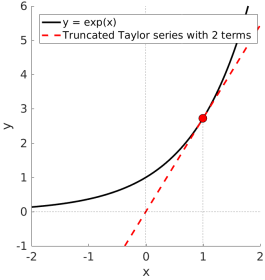
1 term:
Approximate the function as constant

$$f(x) \approx f(a)$$

Truncated Taylor series: geometrical interpretation

Example with function: $x \mapsto \exp(x)$ **and center point** $a = 1$





1 term:
Approximate the function as constant

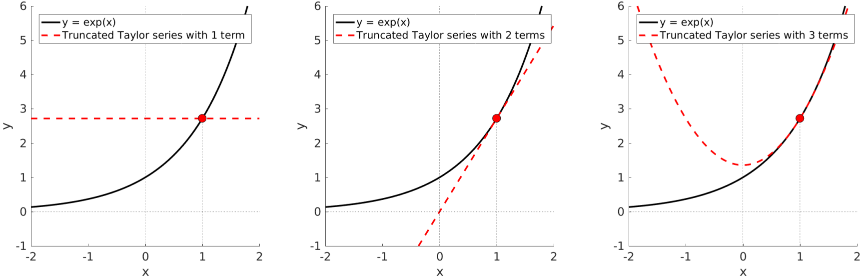
$$f(x) \approx f(a)$$

2 terms:
Approximate the function with its tangent at $x = a$

$$f(x) \approx f(a) + f'(a)(x - a)$$

Truncated Taylor series: geometrical interpretation

Example with function: $x \mapsto \exp(x)$ **and center point** $a = 1$



1 term:
Approximate the function as constant

$$f(x) \approx f(a)$$

2 terms:
Approximate the function with its tangent at $x = a$

$$f(x) \approx f(a) + f'(a)(x - a)$$

More terms:
Add curvature to the approximation

"Classic" Taylor series

Maclaurin series: Taylor series with center point $a = 0$

$$\exp(x) = 1 + \sum_{k=1}^{\infty} \frac{x^k}{k!}$$

$$\ln(1 - x) = -\sum_{k=1}^{\infty} \frac{x^k}{k} \qquad \ln(1 + x) = \sum_{k=1}^{\infty} \frac{(-1)^{k+1} x^k}{k}$$

$$\cos(x) = 1 + \sum_{k=1}^{\infty} \frac{(-1)^k x^{2k}}{(2k)!} \qquad \sin(x) = \sum_{k=0}^{\infty} \frac{(-1)^k x^{2k+1}}{(2k+1)!}$$

$$\cosh(x) = 1 + \sum_{k=1}^{\infty} \frac{x^{2k}}{(2k)!} \qquad \sinh(x) = \sum_{k=0}^{\infty} \frac{x^{2k+1}}{(2k+1)!}$$

Taylor series: practice question

Consider an object falling down vertically. At time $t = t_0 = 1$ s:

- it is **located** at height $z(t = t_0) = 35$ m
- its vertical **velocity** is $v(t = t_0) = -4 \text{ m s}^{-1}$
  (the minus sign indicates that the velocity is downward)
- its vertical **acceleration** is $a(t = t_0) = -10 \text{ m s}^{-2}$
  (the minus sign indicates that the acceleration is downward)

Estimate the height of the item at $t = 3$ s, **using a Taylor series with 1 term**

Consider an object falling down vertically. At time $t = t_0 = 1$ s:

- it is **located** at height $z(t = t_0) = 35$ m
- its vertical **velocity** is $v(t = t_0) = -4 \text{ m s}^{-1}$
  (the minus sign indicates that the velocity is downward)
- its vertical **acceleration** is $a(t = t_0) = -10 \text{ m s}^{-2}$
  (the minus sign indicates that the acceleration is downward)

Estimate the height of the item at $t = 3$ s, **using a Taylor series with 1 term**

$$\begin{aligned} h(t) &\approx h(t_0) \\ h(t = 3\text{ s}) &\approx h(t_0) \\ &= 35 \text{ m} \end{aligned}$$

This approximation is not very good: it indicates that the object has not moved!

Taylor series: practice question

Consider an object falling down vertically. At time $t = t_0 = 1$ s:

- it is **located** at height $z(t = t_0) = 35$ m
- its vertical **velocity** is $v(t = t_0) = -4 \mathrm{~m~s^{-1}}$
  (the minus sign indicates that the velocity is downward)
- its vertical **acceleration** is $a(t = t_0) = -10 \mathrm{~m~s^{-2}}$
  (the minus sign indicates that the acceleration is downward)

Estimate the height of the item at $t = 3$ s, **using a Taylor series with 2 terms**

Taylor series: practice question

Consider an object falling down vertically. At time $t = t_0 = 1$ s:

- it is **located** at height $z(t = t_0) = 35$ m
- its vertical **velocity** is $v(t = t_0) = -4 \mathrm{~m~s^{-1}}$
  (the minus sign indicates that the velocity is downward)
- its vertical **acceleration** is $a(t = t_0) = -10 \mathrm{~m~s^{-2}}$
  (the minus sign indicates that the acceleration is downward)

Estimate the height of the item at $t = 3$ s, **using a Taylor series with 2 terms**

$$h(t) \approx h(t_0) + h'(t_0)(t - t_0)$$

Consider an object falling down vertically. At time $t = t_0 = 1$ s:

- it is **located** at height $z(t = t_0) = 35$ m
- its vertical **velocity** is $v(t = t_0) = -4 \text{ m s}^{-1}$
  (the minus sign indicates that the velocity is downward)
- its vertical **acceleration** is $a(t = t_0) = -10 \text{ m s}^{-2}$
  (the minus sign indicates that the acceleration is downward)

Estimate the height of the item at $t = 3$ s, **using a Taylor series with 2 terms**

$$\begin{aligned} h(t) &\approx h(t_0) + h'(t_0)(t - t_0) \\ &= h(t_0) + v(t_0)(t - t_0) \end{aligned}$$

Taylor series: practice question

Consider an object falling down vertically. At time $t = t_0 = 1$ s:

- it is **located** at height $z(t = t_0) = 35$ m
- its vertical **velocity** is $v(t = t_0) = -4\ \mathrm{m\,s^{-1}}$ (the minus sign indicates that the velocity is downward)
- its vertical **acceleration** is $a(t = t_0) = -10\ \mathrm{m\,s^{-2}}$ (the minus sign indicates that the acceleration is downward)

Estimate the height of the item at $t = 3$ s, **using a Taylor series with 2 terms**

$$
\begin{aligned}
h(t) &\approx h(t_0) + h'(t_0)(t - t_0) \\
&= h(t_0) + v(t_0)(t - t_0) \\
h(t = 3\ \mathrm{s}) &\approx 35\ \mathrm{m} - 4\ \mathrm{m\,s^{-1}} \times (3\ \mathrm{s} - 1\ \mathrm{s}) \\
&= 27\ \mathrm{m}
\end{aligned}
$$

Taylor series: practice question

Consider an object falling down vertically. At time $t = t_0 = 1$ s:

- it is **located** at height $z(t = t_0) = 35$ m
- its vertical **velocity** is $v(t = t_0) = -4 \mathrm{~m~s^{-1}}$
  (the minus sign indicates that the velocity is downward)
- its vertical **acceleration** is $a(t = t_0) = -10 \mathrm{~m~s^{-2}}$
  (the minus sign indicates that the acceleration is downward)

Estimate the height of the item at $t = 3$ s, **using a Taylor series with 3 terms**

Taylor series: practice question

Consider an object falling down vertically. At time $t = t_0 = 1$ s:

- it is **located** at height $z(t = t_0) = 35$ m
- its vertical **velocity** is $v(t = t_0) = -4 \mathrm{~m~s^{-1}}$
  (the minus sign indicates that the velocity is downward)
- its vertical **acceleration** is $a(t = t_0) = -10 \mathrm{~m~s^{-2}}$
  (the minus sign indicates that the acceleration is downward)

Estimate the height of the item at $t = 3$ s, **using a Taylor series with 3 terms**

$$h(t) \approx h(t_0) + h'(t_0)(t - t_0) + \frac{1}{2}h''(t_0)(t - t_0)^2$$

Taylor series: practice question

Consider an object falling down vertically. At time $t = t_0 = 1$ s:

- it is **located** at height $z(t = t_0) = 35$ m
- its vertical **velocity** is $v(t = t_0) = -4 \mathrm{~m~s^{-1}}$
  (the minus sign indicates that the velocity is downward)
- its vertical **acceleration** is $a(t = t_0) = -10 \mathrm{~m~s^{-2}}$
  (the minus sign indicates that the acceleration is downward)

Estimate the height of the item at $t = 3$ s, **using a Taylor series with 3 terms**

$$
\begin{aligned}
h(t) &\approx h(t_0) + h'(t_0)(t - t_0) + \frac{1}{2}h''(t_0)(t - t_0)^2 \\
&= h(t_0) + v(t_0)(t - t_0) + \frac{1}{2}a(t_0)(t - t_0)^2
\end{aligned}
$$

Taylor series: practice question

Consider an object falling down vertically. At time $t = t_0 = 1$ s:

- it is **located** at height $z(t = t_0) = 35$ m
- its vertical **velocity** is $v(t = t_0) = -4 \mathrm{~m~s^{-1}}$
  (the minus sign indicates that the velocity is downward)
- its vertical **acceleration** is $a(t = t_0) = -10 \mathrm{~m~s^{-2}}$
  (the minus sign indicates that the acceleration is downward)

Estimate the height of the item at $t = 3$ s, **using a Taylor series with 3 terms**

$$
\begin{aligned}
h(t) &\approx h(t_0) + h'(t_0)(t - t_0) + \frac{1}{2} h''(t_0)(t - t_0)^2 \\
&= h(t_0) + v(t_0)(t - t_0) + \frac{1}{2} a(t_0)(t - t_0)^2 \\
h(t = 3 \mathrm{~s}) &\approx 35 \mathrm{~m} - 4 \mathrm{~m~s^{-1}}(3 \mathrm{~s} - 1 \mathrm{~s}) - \frac{1}{2} \times 10 \mathrm{~m~s^{-2}} \times (3 \mathrm{~s} - 1 \mathrm{~s})^2 \\
&= 7 \mathrm{~m}
\end{aligned}
$$

E7 final programming project: objectives

Objective: Write a function that decides where your spaceship should go, depending on

- Location of scrap (worth points if picked up)
- Location of slow down areas (nebulae)
- Location of impenetrable areas
- Location and type of ghosts

E7 final programming project: objectives

Objective: Write a function that decides where your spaceship should go, depending on

- Location of scrap (worth points if picked up)
- Location of slow down areas (nebulae)
- Location of impenetrable areas
- Location and type of ghosts

Learning objectives:

E7 final programming project: objectives

Objective: Write a function that decides where your spaceship should go, depending on

- Location of scrap (worth points if picked up)
- Location of slow down areas (nebulae)
- Location of impenetrable areas
- Location and type of ghosts

Learning objectives:

- **Use many of the concepts taught in E7, in one single program**

E7 final programming project: objectives

Objective: Write a function that decides where your spaceship should go, depending on

- Location of scrap (worth points if picked up)
- Location of slow down areas (nebulae)
- Location of impenetrable areas
- Location and type of ghosts

Learning objectives:

- **Use many of the concepts taught in E7, in one single program**
- **Design your own algorithm, make your own choices, prioritize**
  - How to gather scrap efficiently?
  - Should I avoid slow down areas? Try to go through?
  - How to avoid ghosts?
  - ...

E7 final programming project: objectives

Objective: Write a function that decides where your spaceship should go, depending on

- Location of scrap (worth points if picked up)
- Location of slow down areas (nebulae)
- Location of impenetrable areas
- Location and type of ghosts

Learning objectives:

- **Use many of the concepts taught in E7, in one single program**
- **Design your own algorithm, make your own choices, prioritize**
  - How to gather scrap efficiently?
  - Should I avoid slow down areas? Try to go through?
  - How to avoid ghosts?
  - ...
- **Write modular code**

  Write separate functions, each performing a specific task

E7 final programming project: objectives

Objective: Write a function that decides where your spaceship should go, depending on

- Location of scrap (worth points if picked up)
- Location of slow down areas (nebulae)
- Location of impenetrable areas
- Location and type of ghosts

Learning objectives:

- **Use many of the concepts taught in E7, in one single program**
- **Design your own algorithm, make your own choices, prioritize**
  - How to gather scrap efficiently?
  - Should I avoid slow down areas? Try to go through?
  - How to avoid ghosts?
  - ...
- **Write modular code**

  Write separate functions, each performing a specific task
- **Work in groups**

E7 final programming project: objectives

Objective: Write a function that decides where your spaceship should go, depending on

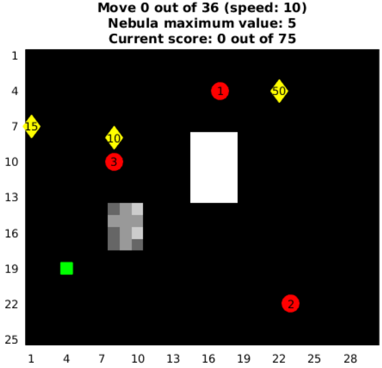
- Location of scrap (worth points if picked up)
- Location of slow down areas (nebulae)
- Location of impenetrable areas
- Location and type of ghosts

Learning objectives:

- **Use many of the concepts taught in E7, in one single program**
- **Design your own algorithm, make your own choices, prioritize**
  - How to gather scrap efficiently?
  - Should I avoid slow down areas? Try to go through?
  - How to avoid ghosts?
  - ...
- **Write modular code**

  Write separate functions, each performing a specific task
- **Work in groups**
- **Have fun!**

E7 final programming project: game engine and visualizer



The function that runs the game will be provided to you:

```
function [game_stats] = e7planets_play(map, player_function)
```

- map: describes the game area (location of scrap, ghosts, ...)
  - We will provide you with sample maps
  - You can create your own maps
- player_function: **the function that you will write!**

E7 final programming project: your task

Write a function with the following header:

```
function [direction] = player_function(map)
```

E7 final programming project: your task

Write a function with the following header:

```
function [direction] = player_function(map)
```

`direction`: where your ship should go next

- `'U'`, `'D'`, `'L'`, `'R'`, or `'.'` (up, down, left, right, or “don’t move”)

E7 final programming project: your task

Write a function with the following header:

```
function [direction] = player_function(map)
```

`direction`: where your ship should go next

- `'U'`, `'D'`, `'L'`, `'R'`, or `'.'` (up, down, left, right, or “don't move”)

`map`: 1×1 `struct` array

E7 final programming project: your task

Write a function with the following header:

```
function [direction] = player_function(map)
```

`direction`: where your ship should go next

- 'U', 'D', 'L', 'R', or '.' (up, down, left, right, or “don't move”)

`map`: 1×1 `struct` array

- `map.grid`: $m \times n$ array of class `double` containing integers
  - A 0 indicates a place where you cannot go
  - A 1 indicates a place you can travel to and from
  - A number greater than 1 indicates a slow-down location. Each time you try to move away from a slow down location (`map.grid(i,j)>1`), you have 1 in `map.grid(i,j)` chances to successfully move away from this location

E7 final programming project: your task

Write a function with the following header:

```
function [direction] = player_function(map)
```

`direction`: where your ship should go next

- `'U'`, `'D'`, `'L'`, `'R'`, or `'.'` (up, down, left, right, or “don’t move”)

`map`: 1×1 `struct` array

- `map.grid`: $m \times n$ array of class `double` containing integers
  - A 0 indicates a place where you cannot go
  - A 1 indicates a place you can travel to and from
  - A number greater than 1 indicates a slow-down location. Each time you try to move away from a slow down location (`map.grid(i,j)>1`), you have 1 in `map.grid(i,j)` chances to successfully move away from this location
- `map.player`: 1×1 `struct` array that indicates your location (previous and current) and your current score

- `map.scrap`: `struct` array that indicates the locations and values of scrap

E7 final programming project: your task

- `map.scrap`: `struct` array that indicates the locations and values of scrap
- `map.ghosts`: `struct` array that indicates the location (previous and current) and type of each ghost. Ghost types:
  - `'random'`: moves randomly
  - `'backandforth'`: moves in a straight line, back and forth
  - `'towardplayer'`: moves toward your spaceship

E7 final programming project: your task

- `map.scrap`: `struct` array that indicates the locations and values of scrap
- `map.ghosts`: `struct` array that indicates the location (previous and current) and type of each ghost. Ghost types:
  - `'random'`: moves randomly
  - `'backandforth'`: moves in a straight line, back and forth
  - `'towardplayer'`: moves toward your spaceship
- `map.remaining_moves`: number of moves you can make before the end of the game

E7 final programming project: your task

- `map.scrap`: `struct` array that indicates the locations and values of scrap
- `map.ghosts`: `struct` array that indicates the location (previous and current) and type of each ghost. Ghost types:
  - `'random'`: moves randomly
  - `'backandforth'`: moves in a straight line, back and forth
  - `'towardplayer'`: moves toward your spaceship
- `map.remaining_moves`: number of moves you can make before the end of the game

You win the game if you pick up all the scrap within the number of allocated moves, and without being caught by a ghost

E7 final programming project: grading

Your function will be graded on **many different maps of varying size and difficulty**, for example:

- Scrap only
- Scrap and slow-down areas
- Scrap and obstacles
- Scrap, slow-down areas, and obstacles
- All of the above, with ghosts

E7 final programming project: grading

Your function will be graded on **many different maps of varying size and difficulty**, for example:

- Scrap only
- Scrap and slow-down areas
- Scrap and obstacles
- Scrap, slow-down areas, and obstacles
- All of the above, with ghosts

There will be **partial credit** if, by the end of the game, you have picked up some of the scrap, but not all

E7 final programming project: grading

Your function will be graded on **many different maps of varying size and difficulty**, for example:

- Scrap only
- Scrap and slow-down areas
- Scrap and obstacles
- Scrap, slow-down areas, and obstacles
- All of the above, with ghosts

There will be **partial credit** if, by the end of the game, you have picked up some of the scrap, but not all

Recommendation: **focus on basic functionality before trying to implement advanced features**