

PEPit.jl tutorial: Accelerated gradient method for smooth strongly convex minimization

John Doe

December 1, 2025

1 Setup

Before we do anything let us load the necessary packages.

```
using PEPit, OrderedCollections, Mosek, MosekTools, Clarabel
```

Let us start with defining an empty PEP, which we will construct step by step.

```
problem = PEP()
```

2 Problem in consideration

Consider the convex minimization problem

$$f_{\star} \triangleq \min_x f(x),$$

where f is L -smooth and μ -strongly convex. For this example, let us take $\mu = 0.1$ and $L = 1$. Let us declare this function type in PEPit next. Let us start with defining the parameters μ and L .

```
μ = 0.1  
L = 1  
param = OrderedDict{"μ" => μ, "L" => L}
```

This defines the set of parameters μ and L to describe the function class. Okay, now time to define the function class itself.

```
func = declare_function!(problem, SmoothStronglyConvexFunction, param; reuse_gradient=true) #  
↳ creates the class of strongly convex and smooth functions with parameters  $\mu = 0.1$  and  
↳  $L = 1$ 
```

3 Algorithm

What we want to study in this tutorial is to compute the a worst-case guarantee for the **fast gradient** method described as follows.

3.1 Algorithm:

For $t \in \{0, \dots, n-1\}$,

$$y_t = x_t + \frac{\sqrt{L} - \sqrt{\mu}}{\sqrt{L} + \sqrt{\mu}}(x_t - x_{t-1})$$

$$x_{t+1} = y_t - \frac{1}{L} \nabla f(y_t)$$

with $x_{-1} := x_0$.

4 Goal

We want to compute the smallest possible $\tau(n, L, \mu)$ such that the guarantee

$$f(x_n) - f_\star \leq \tau(n, L, \mu) \left(f(x_0) - f(x_\star) + \frac{\mu}{2} \|x_0 - x_\star\|^2 \right),$$

is valid, where x_n is the output of the **fast gradient** method, and where x_\star is the minimizer of f . Sometimes we say that $(f(x_0) - f(x_\star) + \frac{\mu}{2} \|x_0 - x_\star\|^2)$ is the initial condition for this PEP. In short, for given values of n, L and μ , $\tau(n, L, \mu)$ is computed as the worst-case value of $f(x_n) - f_\star$ when $f(x_0) - f(x_\star) + \frac{\mu}{2} \|x_0 - x_\star\|^2 \leq 1$.

5 Construct the PEP step by step

Let us implement the algorithm in PEPit now. For this introductory example we will let $n = 2$.

```
n = 2
```

5.1 Define x_0, x_\star, f_\star

We start with defining the initial point $x_0 \equiv \text{x0}$, optimal point $x_\star \equiv \text{xs}$, and the optimal objective value $f_\star \equiv \text{fs}$.

```
xs = stationary_point!(func) # creates x_\star
fs = value!(func, xs) # computes f_\star = f(x_\star)
x0 = set_initial_point!(problem) # creates x_0
```

5.2 Define initial condition

The next step is now define our initial condition, which is $f(x_0) - f(x_\star) + \frac{\mu}{2} \|x_0 - x_\star\|^2 \leq 1$.

```
set_initial_condition!(problem, value!(func, x0) - fs + \mu / 2 * (x0 - xs)^2 <= 1) # creates
↪ the initial condition f(x_0) - f(x_\star) + \frac{\mu}{2} \|x_0 - x_\star\|^2 \leq 1
```

5.3 Describe the algorithm

Great, now it is time to implement the *fast gradient method* in a loop.

```
κ = μ/L

x_new, y = x0, x0
for i in 1:n
    global x_new, y # mark both as globals in this loop
    x_old = x_new
    x_new = y - 1 / L * gradient!(func, y)
    y = x_new + (1 - sqrt(κ)) / (1 + sqrt(κ)) * (x_new - x_old)
end
```

So when the loop ends `x_new` contains x_n . Let us now evaluate $f(x_n)$.

```
f_n = value!(func, x_new) # computes f(x_n)
```

Let us now tell the PEP that our performance measure is $f(x_N) - f_\star$.

```
set_performance_metric!(problem, value!(func, x_new) - fs) # set performance metric to
↪ f(x_n) - f_*
```

5.4 Solve the PEP

We have everything to build up the PEP. Now time to solve!

```
τ_PEPit = solve!(problem; solver = Mosek.Optimizer, verbose=true) # we can use other SDP
↪ solvers such as Clarabel, via 'solver = Clarabel.Optimizer'
```

5.5 Comparison with theoretical results

If everything went well, we should get $\tau_{\text{PEPit}} \approx 0.34760221803672986$ or something very close to it. What does the theory say in this regard? The following **upper** guarantee can be found in [1, Corollary 4.15]:

$$f(x_n) - f_\star \leq \left(1 - \sqrt{\frac{\mu}{L}}\right)^n \left(f(x_0) - f(x_\star) + \frac{\mu}{2} \|x_0 - x_\star\|^2\right).$$

Let us compute the theoretical value of τ .

```
τ_Theory = (1 - sqrt(κ))^n

@info "Ⓜ PEPit guarantee: f(x_n)-f_* <= $(round(τ_PEPit, digits=6)) (f(x_0) - f(x_*) +
↪ μ/2*||x_0 - x_*||^2)"

@info "Ⓜ Theoretical guarantee: f(x_n)-f_* <= $(round(τ_Theory, digits=6)) (f(x_0) - f(x_*) +
↪ μ/2*||x_0 - x_*||^2)"
```

We should see the output:

```
PEPit guarantee: f(x_n)-f_* <= 0.347602 (f(x_0) - f(x_*) + mu/2*||x_0 - x_*||^2)
```

```
Theoretical guarantee: f(x_n)-f_* <= 0.467544 (f(x_0) - f(x_*) + mu/2*||x_0 - x_*||^2)
```

So the gurantee that we found via PEPit is tighter than the theoretical guarantee!

References:

[1] A. d'Aspremont, D. Scieur, A. Taylor (2021). Acceleration Methods. Foundations and Trends in Optimization: Vol. 5, No. 1-2.