AGIMUS WINTER SCHOOL 2023

OPTIMIZATION AND OPTIMAL CONTROL II

Wilson Jallet

LAAS-CNRS, Gepetto team / Inria, Willow team

Overview

1. Constrained optimization

   A kind refresher

   ProxQP – AL methods applied to QPs

   Augmented Lagrangians for general NLPs

2. Constrained trajectory optimization

   Problem definition

   Augmented Lagrangian trajectory optimization with ProxDDP

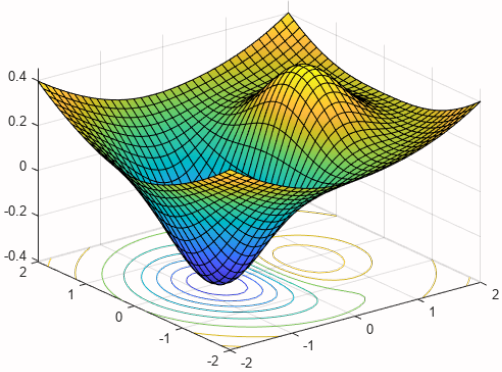
Foreword

The goal of this presentation is to (re)familiarize yourself with concepts from **CONSTRAINED OPTIMIZATION** and its difficulties.

We will talk of **NONLINEAR PROGRAMS** (NLPs) in general and apply the concepts of proximal methods to tackle them, first in the quadratic programming and later for optimal control.

Constrained optimization

A kind refresher



Unconstrained optimization: only needs an objective function $c : \mathbb{R}^{n_x} \to \mathbb{R}$. The problem is simply:

$$\underset{x \in \mathbb{R}^{n_x}}{\text{minimize}}\, c(x). \tag{1}$$

A point $x^\star \in \mathbb{R}^{n_x}$ is a **LOCAL MINIMIZER** if

$$\text{for all } x' \text{ in a neighborhood of } x^\star,\ f(x^\star) \leq f(x')$$

and a *strict* local min. if $f(x^\star) < f(x')$ for $x' \neq x^\star$.

Remark

$c(x) \in \mathbb{R} \Rightarrow$ there are no implicit constraints (as introduced in Adrien's talk)

Recall – Global minima and convexity

A point $x^\star$ is a **GLOBAL MINIMUM** if for *all* $x' \in \mathbb{R}^n$, $f(x^\star) \leq f(x')$.

When does local imply global?

Recall – Global minima and convexity

A point $x^\star$ is a **GLOBAL MINIMUM** if for *all* $x' \in \mathbb{R}^n$, $f(x^\star) \leq f(x')$.

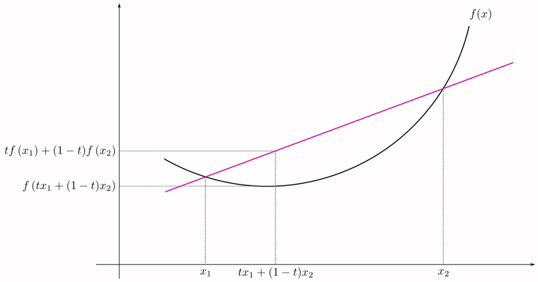
When does local imply global? When the function f is **CONVEX**:

Definition (Convexity)

f is called *convex* when for any x, y and $t \in [0,1]$,

$$f(tx + (1-t)y) \leq tf(x) + (1-t)f(y).$$

Strictly convex when for $x \neq y$ and $t \in (0,1)$, the inequality is strict.



Recall – Global minima and convexity

A point $x^\star$ is a **GLOBAL MINIMUM** if for *all* $x' \in \mathbb{R}^n$, $f(x^\star) \leq f(x')$.

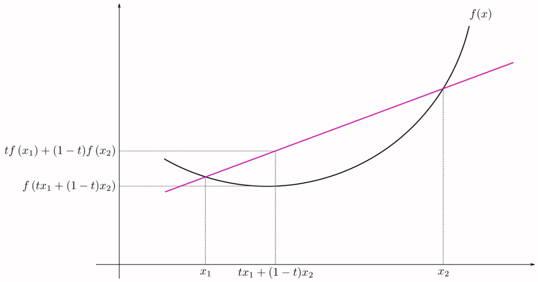
When does local imply global? When the function f is **CONVEX**:

Definition (Convexity)

f is called *convex* when for any x, y and $t \in [0, 1]$,

$$f(tx + (1-t)y) \leq tf(x) + (1-t)f(y).$$

Strictly convex when for $x \neq y$ and $t \in (0, 1)$, the inequality is strict.



Alternative characterization: if f has second derivatives, when $\nabla^2 f \succeq 0$ ($\succ 0$ for *strict* convexity).

Question: how do we know a point $x^\star \in \mathbb{R}^{n_x}$ is a (local) minimizer?

STATIONARITY CONDITIONS: if $x^\star$ is a *local optimum*, then $x^\star$ **is an optimum along any line**:

$$\text{for all } v \in \mathbb{R}^{n_x}, \ \frac{d}{dt}(c(x^\star + tv))\Big|_{t=0} = \langle v, \nabla c(x^\star) \rangle = 0, \tag{2}$$

i.e. the *first-order condition*:

$$\boxed{\nabla c(x^\star) = 0.} \tag{3}$$

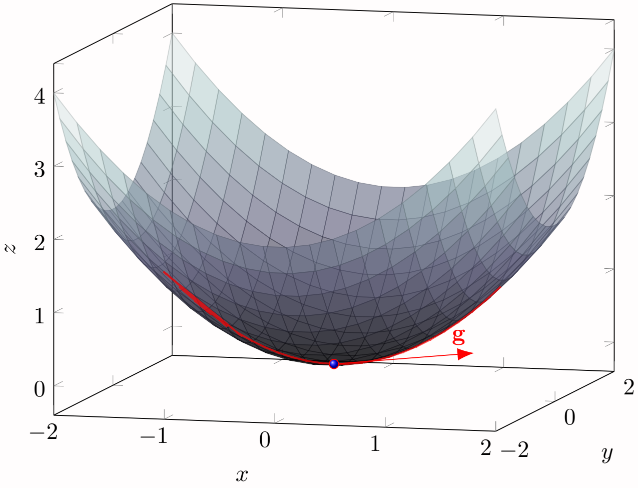


Figure 1: At the minimum, the tangent vectors to the graph $\mathbf{g}$ are flat – i.e. they are all of the form $(g_x, g_y, 0)$.

Constrained optimization

Consider the (smooth) *constrained* minimization problem

$$\min_{x \in \mathbb{R}^{n_x}} c(x) \tag{4a}$$

$$\text{s.t. } g(x) = 0 \tag{4b}$$

$$h(x) \leq 0. \tag{4c}$$

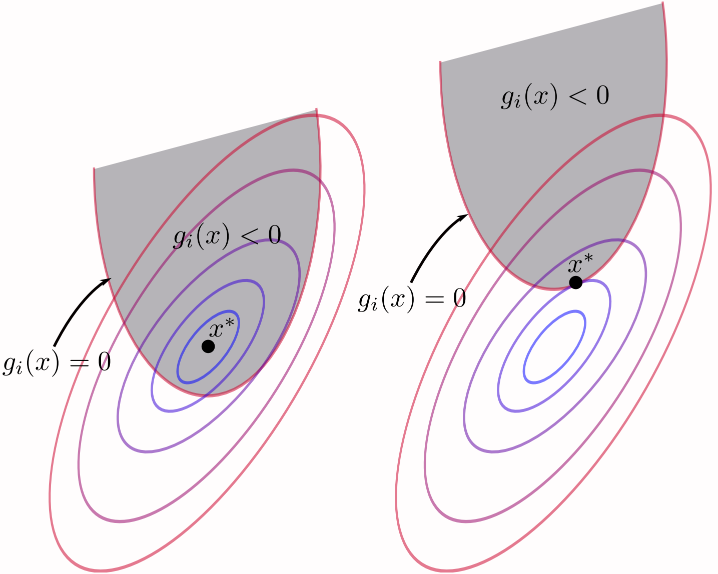


diagram source: Wikipedia

AGIMUS
INNOVATIVE ROBOTICS FOR AGILE PRODUCTION

Necessary conditions

Given by the **KKT CONDITIONS**: a point $x^\star \in \mathbb{R}^{n_x}$ is a **LOCAL MINIMIZER** if there are **LAGRANGE MULTIPLIERS** $(y^\star, z^\star) \in \mathbb{R}^{n_g} \times \mathbb{R}_+^{n_h}$ satisfying

$$\nabla c(x^\star) + \partial_x g(x^\star)^\top y^\star + \partial_x h(x^\star)^\top z^\star = 0 \qquad \text{(stationarity)} \tag{5a}$$

$$g(x^\star) = 0 \qquad \text{(eq. constraint)} \tag{5b}$$

$$h(x^\star) \leq 0 \qquad \text{(ineq. constraint)} \tag{5c}$$

$$h(x^\star) \odot z^\star = 0 \ (h_i z_i = 0) \qquad \text{(complementarity)} \tag{5d}$$

Necessary conditions

Given by the **KKT CONDITIONS**: a point $x^\star \in \mathbb{R}^{n_x}$ is a **LOCAL MINIMIZER** if there are **LAGRANGE MULTIPLIERS** $(y^\star, z^\star) \in \mathbb{R}^{n_g} \times \mathbb{R}^{n_h}_{+}$ satisfying

$$\nabla c(x^\star) + \partial_x g(x^\star)^\top y^\star + \partial_x h(x^\star)^\top z^\star = 0 \qquad \text{(stationarity)} \tag{5a}$$

$$g(x^\star) = 0 \qquad \text{(eq. constraint)} \tag{5b}$$

$$h(x^\star) \leq 0 \qquad \text{(ineq. constraint)} \tag{5c}$$

$$h(x^\star) \odot z^\star = 0 \ (h_i z_i = 0) \qquad \text{(complementarity)} \tag{5d}$$

Equation (5a) above is the gradient of the classical **LAGRANGIAN FUNCTION** (Rockafellar 1997)

$$\mathscr{L}(x, y, z) = c(x) + y^\top g(x) + z^\top h(x). \tag{6}$$

Necessary conditions

Given by the **KKT CONDITIONS**: a point $x^\star \in \mathbb{R}^{n_x}$ is a **LOCAL MINIMIZER** if there are **LAGRANGE MULTIPLIERS** $(y^\star, z^\star) \in \mathbb{R}^{n_g} \times \mathbb{R}_+^{n_h}$ satisfying

$$
\begin{aligned}
&\nabla c(x^\star) + \partial_x g(x^\star)^\top y^\star + \partial_x h(x^\star)^\top z^\star = 0 && \text{(stationarity)} && (5a)\\
&g(x^\star) = 0 && \text{(eq. constraint)} && (5b)\\
&h(x^\star) \leq 0 && \text{(ineq. constraint)} && (5c)\\
&h(x^\star) \odot z^\star = 0 \ (h_i z_i = 0) && \text{(complementarity)} && (5d)
\end{aligned}
$$

Equation (5a) above is the gradient of the classical **LAGRANGIAN FUNCTION** (Rockafellar 1997)

$$
\mathscr{L}(x, y, z) = c(x) + y^\top g(x) + z^\top h(x). \tag{6}
$$

(5d) are called the **COMPLEMENTARITY CONDITIONS**. The set of i such that $z_i^\star > 0$ $(h_i(x^\star) = 0)$ is called the **ACTIVE SET OF CONSTRAINTS**.

Some things which are NLPs:

- quadratic programs (QPs), among which linear-quadratic (LQ) control problems
- contact problems (see Quentin's stuff)
- collision detection (talk to Louis)
- inverse kinematics
- others?...

Some things which are NLPs:

- quadratic programs (QPs), among which linear-quadratic (LQ) control problems
- contact problems (see Quentin's stuff)
- collision detection (talk to Louis)
- inverse kinematics
- others?...

Convex?

- if f, h is convex, and g is affine (sorry)

As Adrien pointed out, **general nonlinear programming is hard**.

Many methods exist:

- straight **sequential quadratic programming** (SQP), solving a cascade of inequality-QPs with linesearch/filter/trust-region strategies, see SNOPT (Gill *et al.* 2002)
- **interior-point methods**: add barrier for inequalities then move to equality-SQP, see IPOPT (Wächter and Biegler 2006)
- **augmented Lagrangian** methods, with second-order approaches e.g. LANCELOT (A. R. Conn *et al.* 2010)

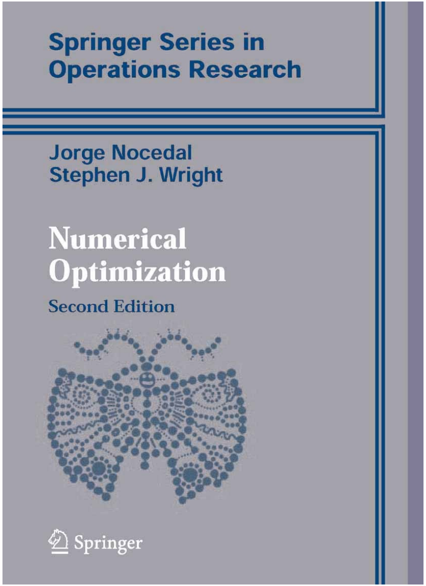


Figure 2: **The holy book:** *Numerical Optimization* (Nocedal and Wright 2006)

Constrained optimization

ProxQP – AL methods applied to QPs

Equality-constrained QPs (EQPs)

The problem. Let $Q \in \mathbf{S}_n^+(\mathbb{R})$, $q \in \mathbb{R}^n$, $A \in \mathbb{R}^{m \times n}$ and $b \in \mathbb{R}^m$. We consider the simple equality-constrained QP

$$\begin{aligned} \min_x \ & \frac{1}{2} x^\top Q x + q^\top x \\ \text{s.t. } & Ax + b = 0 \end{aligned} \tag{EQP}$$

Lagrangian:

$$\mathcal{L}(x, y) = \tfrac{1}{2} x^\top Q x + q^\top x + y^\top (Ax + b). \tag{7}$$

Equality-constrained QPs (EQPs)

The problem. Let $Q \in \mathbf{S}_n^+(\mathbb{R})$, $q \in \mathbb{R}^n$, $A \in \mathbb{R}^{m\times n}$ and $b \in \mathbb{R}^m$. We consider the simple equality-constrained QP

$$\begin{aligned} \min_x \ & \frac{1}{2} x^\top Q x + q^\top x \\ \text{s.t. } & Ax + b = 0 \end{aligned} \tag{EQP}$$

Lagrangian:

$$\mathcal{L}(x, y) = \tfrac{1}{2} x^\top Q x + q^\top x + y^\top (Ax + b). \tag{7}$$

KKT conditions. Very classically:

$$\begin{bmatrix} Q & A^\top \\ A & \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = - \begin{bmatrix} q \\ b \end{bmatrix}. \tag{8}$$

Unique solution iff matrix is invertible.

Unique solution $(x^\star, y^\star)$ iff KKT matrix $\begin{bmatrix} Q & A^\top \\ A & \end{bmatrix}$ is invertible.

Proposition (see Nocedal and Wright 2006, chap. 16)

The KKT matrix is nonsingular if:

- **LICQ** (*linear independence constraint qualification*) i.e. linear independence of rows of A
- if Z basis matrix $\ker(A)$ (i.e. Z full rank, $AZ = 0$), then $Z^\top QZ \succ 0$.

Unique solution $(x^\star, y^\star)$ iff KKT matrix $\begin{bmatrix} Q & A^\top \\ A & \end{bmatrix}$ is invertible.

Proposition (see Nocedal and Wright 2006, chap. 16)

The KKT matrix is nonsingular if:

- **LICQ** (*linear independence constraint qualification*) i.e. linear independence of rows of A
- if Z basis matrix $\ker(A)$ (i.e. Z full rank, $AZ = 0$), then $Z^\top Q Z \succ 0$. (*Strict convexity* ($Q \succ 0$) is sufficient[1])

[1] Even required by some solvers e.g. QUADPROG (`https://github.com/quadprog/quadprog`) based on Goldfarb and Idnani 1983 Goldfarb and Idnani 1983

Unique solution $(x^\star, y^\star)$ iff KKT matrix $\begin{bmatrix} Q & A^\top \\ A & \end{bmatrix}$ is invertible.

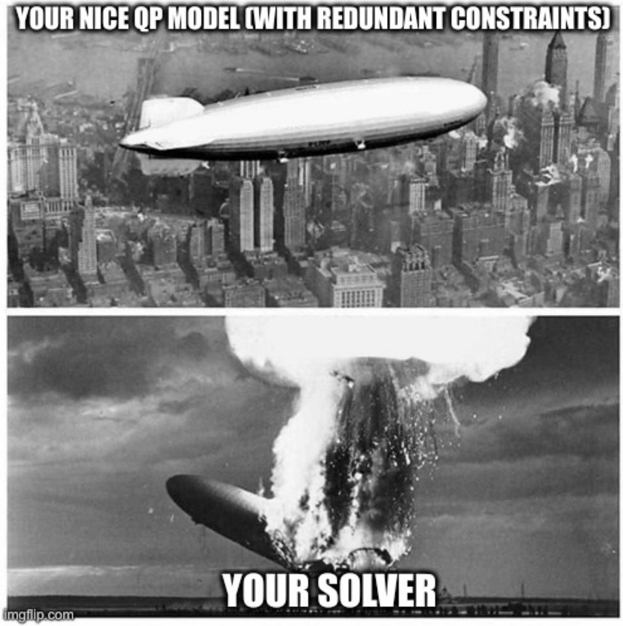
Proposition (see Nocedal and Wright 2006, chap. 16)

The KKT matrix is nonsingular if:

- **LICQ** (*linear independence constraint qualification*) i.e. linear independence of rows of A
- if Z basis matrix $\ker(A)$ (i.e. Z full rank, $AZ = 0$), then $Z^\top Q Z \succ 0$. (*Strict convexity* ($Q \succ 0$) is sufficient[1])

In practice: not very fun! (no redundant constraints)

[1] Even required by some solvers e.g. QUADPROG (`https://github.com/quadprog/quadprog`) based on Goldfarb and Idnani 1983 Goldfarb and Idnani 1983



AGIMUS
INNOVATIVE ROBOTICS FOR AGILE PRODUCTION

Method of multipliers for rank-deficient EQPs

Redundant constraints? Augmented Lagrangians (AL) to the rescue!

The primal way. Let $\mu > 0$. The AL associated with (EQP) is the quadratic

$$\mathcal{L}_\mu(x; y_e) \stackrel{\text{def}}{=} \frac{1}{2}x^\top Qx + q^\top x + y_e^\top (Ax + b) + \tfrac{1}{2\mu}\|Ax + b\|_2^2$$

Method of multipliers for rank-deficient EQPs

Redundant constraints? Augmented Lagrangians (AL) to the rescue!

The primal way. Let $\mu > 0$. The AL associated with (EQP) is the quadratic

$$\begin{aligned}\mathcal{L}_\mu(x; y_e) &\stackrel{\text{def}}{=} \frac{1}{2}x^\top Qx + q^\top x + y_e^\top(Ax+b) + \tfrac{1}{2\mu}\|Ax+b\|_2^2 \\ &= -\underbrace{\min_y \left\{-\mathcal{L}(x,y) + \tfrac{\mu}{2}\|y-y_e\|_2^2\right\}}_{\textit{proximal!}}\end{aligned} \tag{9}$$

Method of multipliers for rank-deficient EQPs

Redundant constraints? Augmented Lagrangians (AL) to the rescue!

The primal way. Let $\mu > 0$. The AL associated with (EQP) is the quadratic

$$\begin{aligned}\mathcal{L}_\mu(x; y_e) &\stackrel{\text{def}}{=} \frac{1}{2}x^\top Qx + q^\top x + y_e^\top(Ax+b) + \tfrac{1}{2\mu}\|Ax+b\|_2^2 \\ &= \underbrace{-\min_y \left\{-\mathcal{L}(x,y) + \tfrac{\mu}{2}\|y - y_e\|_2^2\right\}}_{\textit{proximal!}}\end{aligned} \tag{9}$$

Method of multipliers. Minimum given by $\nabla_x \mathcal{L}_\mu(x^+; y_e) = 0$ i.e.

$$(Q + \tfrac{1}{\mu}A^\top A)x^+ = -[q + A^\top(y_e + \tfrac{1}{\mu}b)] \tag{10}$$

and dual step $y^+ = y_e + \frac{1}{\mu}(Ax^+ + b)$.

Set $x \leftarrow x^+$, $y_e \leftarrow y^+$, **rinse and repeat.**

Method of multipliers for rank-deficient EQPs

Redundant constraints? Augmented Lagrangians (AL) to the rescue!

The primal way. Let $\mu > 0$. The AL associated with (EQP) is the quadratic

$$\begin{aligned}\mathcal{L}_\mu(x; y_e) &\overset{\text{def}}{=} \frac{1}{2}x^\top Qx + q^\top x + y_e^\top(Ax+b) + \tfrac{1}{2\mu}\|Ax+b\|_2^2 \\ &= -\underbrace{\min_y \left\{-\mathcal{L}(x,y) + \tfrac{\mu}{2}\|y-y_e\|_2^2\right\}}_{\textit{proximal!}}\end{aligned} \tag{9}$$

Method of multipliers. Minimum given by $\nabla_x \mathcal{L}_\mu(x^+; y_e) = 0$ i.e.

$$(Q + \tfrac{1}{\mu}A^\top A)x^+ = -[q + A^\top(y_e + \tfrac{1}{\mu}b)] \tag{10}$$

and dual step $y^+ = y_e + \frac{1}{\mu}(Ax^+ + b)$.
Set $x \leftarrow x^+$, $y_e \leftarrow y^+$, **rinse and repeat.**

Caveat: *bad* numerical conditioning (matrix eigenvalues might span a large range of values e.g. 10^{-6} to 10^6)

Primal-dual/saddle-point view. Introduces a regularized KKT matrix:

$$\begin{bmatrix} Q & A^\top \\ A & -\mu I \end{bmatrix} \begin{bmatrix} x^+ \\ y^+ \end{bmatrix} = - \begin{bmatrix} q \\ b + \mu y_e \end{bmatrix} \tag{11}$$

Remark

- μ controls convergence speed $\rightarrow$ lower is faster (but less stable)
- clever heuristics for $\{\mu_k\}$ for good compromises e.g. BCL (A. Conn *et al.* 1991)

Further explored in the practical session!

A link through linear algebra with **Schur complements**:

$$Q + \frac{1}{\mu} A^\top A \xLeftrightarrow{\text{Schur compl.}} \begin{bmatrix} Q & A^\top \\ A & -\mu I \end{bmatrix} \xLeftrightarrow{\text{Schur compl.}} \mu I + A Q^{-1} A^\top \tag{12}$$

2nd variant is similar to Goldfarb and Idnani 1983, also used in Carpentier *et al.* 2021 (RSS).

Robotics: Science and Systems 2021
Held Virtually, July 12–16, 2021

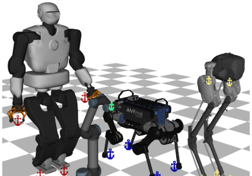
Proximal and Sparse Resolution of Constrained Dynamic Equations

Justin Carpentier
Inria, École normale supérieure
CNRS, PSL Research University
75005 Paris, France
Email: justin.carpentier@inria.fr

Rohan Budhiraja
Inria Paris
75012 Paris, France
Email: rohan.budhiraja@inria.fr

Nicolas Mansard
LAAS-CNRS, ANITI
University of Toulouse
31400 Toulouse, France
Email: nicolas.mansard@laas.fr

***Abstract*—Control of robots with kinematic constraints like loop-closure constraints or interactions with the environment requires solving the underlying constrained dynamics equations of motion. Several approaches have been proposed so far in the literature to solve these constrained optimization problems, for instance by either taking advantage in part of the sparsity of the kinematic tree or by considering an explicit formulation of the constraints in the problem resolution. Yet, not all the constraints allow an explicit formulation and in general, approaches of the state of the art suffer from singularity issues, especially in the context of redundant or singular constraints. In this paper, we propose a unified approach to solve forward dynamics equations involving constraints in an efficient, generic and robust manner. To this aim, we first (i) propose a proximal formulation of the constrained dynamics which converges to an optimal solution in the least-square sense even in the presence of singularities.**



AGIMUS
INNOVATIVE ROBOTICS FOR AGILE PRODUCTION

Inequality-constrained QPs

A (slightly?) harder problem:

$$\min_x \frac{1}{2} x^\top Q x + q^\top x \tag{13a}$$

$$\text{s.t. } Ax + b = 0 \tag{13b}$$

$$Cx + u \leq 0 \tag{13c}$$

Inequality-constrained QPs

A (slightly?) harder problem:

$$\min_x \frac{1}{2} x^\top Q x + q^\top x \tag{13a}$$
$$\text{s.t. } Ax + b = 0 \tag{13b}$$
$$Cx + u \leq 0 \tag{13c}$$

Actually **WAY HARDER**.

KKT CONDITIONS are like before, plus the complementarity:

$$Qx + q + A^\top y + C^\top z = 0 \tag{14a}$$
$$Ax + b = 0 \tag{14b}$$
$$Cx + u \leq 0 \tag{14c}$$
$$z \odot [Cx + u] = 0 \tag{14d}$$

Inequality-constrained QPs

A (slightly?) harder problem:

$$\min_x \frac{1}{2} x^\top Q x + q^\top x \tag{13a}$$
$$\text{s.t. } Ax + b = 0 \tag{13b}$$
$$Cx + u \leq 0 \tag{13c}$$

KKT CONDITIONS are like before, plus the complementarity:

$$Qx + q + A^\top y + C^\top z = 0 \tag{14a}$$
$$Ax + b = 0 \tag{14b}$$
$$Cx + u \leq 0 \tag{14c}$$
$$z \odot [Cx + u] = 0 \tag{14d}$$

Actually **WAY HARDER**. Many methods employed for this:

- dual method (*strictly convex*) AKA Goldfarb and Idnani 1983 AKA quadprog
- solve EQP + ADMM (see the OSQP solver (Stellato *et al.* 2020))
- active-set search ~~sorcery~~ (solver: qpOASES (Ferreau *et al.* 2014))
- and AL! See QPALM (Hermans *et al.* 2019), QPDO (De Marchi 2022) and ours, **ProxQP** (Bambade *et al.* 2023)

AGIMUS
INNOVATIVE ROBOTICS FOR AGILE PRODUCTION

AL for inequality-constrained problems

(Generalized) AL function. (see Rockafellar 1976)

$$\mathcal{L}_\mu(x; y_e, z_e) = \frac{1}{2}x^\top Qx + q^\top x + \underbrace{y_e^\top (Ax+b) + \frac{1}{2\mu}\|Ax+b\|_2^2}_{\text{equality penalty}} + \underbrace{\frac{1}{2\mu}\|[Cx+u+\mu z_e]_+\|_2^2 - \frac{\mu}{2}\|z_e\|_2^2}_{\text{inequality penalty}}. \tag{15}$$

AGIMUS INNOVATIVE ROBOTICS FOR AGILE PRODUCTION

AL for inequality-constrained problems

(Generalized) AL function. (see Rockafellar 1976)

$$\mathcal{L}_\mu(x; y_e, z_e) = \frac{1}{2}x^\top Qx + q^\top x + \underbrace{y_e^\top (Ax + b) + \frac{1}{2\mu}\|Ax + b\|_2^2}_{\text{equality penalty}} + \underbrace{\frac{1}{2\mu}\|[Cx + u + \mu z_e]_+\|_2^2 - \frac{\mu}{2}\|z_e\|_2^2}_{\text{inequality penalty}}. \tag{15}$$

Terrible news!

- *Not* quadratic anymore – just piecewise.
- **No closed-form minimum.**
- **Not even smooth!**

AL for inequality-constrained problems

(Generalized) AL function. (see Rockafellar 1976)

$$\mathcal{L}_\mu(x; y_e, z_e) = \frac{1}{2}x^\top Qx + q^\top x + \underbrace{y_e^\top(Ax+b) + \frac{1}{2\mu}\|Ax+b\|_2^2}_{\text{equality penalty}} + \underbrace{\frac{1}{2\mu}\|[Cx+u+\mu z_e]_+\|_2^2 - \frac{\mu}{2}\|z_e\|_2^2}_{\text{inequality penalty}}. \tag{15}$$

Terrible news!

- *Not* quadratic anymore – just piecewise.
- **No closed-form minimum.**
- **Not even smooth!**

Methods such as **ProxQP** and QPALM $\rightarrow$ **inexact minimization using semi-smooth Newton methods** (not covered in this session).

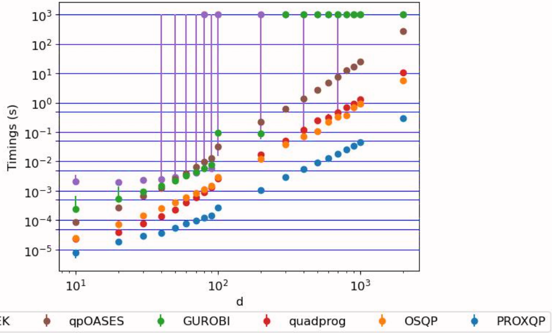
In general, these methods are **difficult to implement**, especially with **performance** in mind.

In general, these methods are **difficult to implement**, especially with **performance** in mind.

Try our solver!

ProxSuite

THE ADVANCED PROXIMAL OPTIMIZATION TOOLBOX



`conda install -c conda-forge proxsuite`

AGIMUS

INNOVATIVE ROBOTICS FOR AGILE PRODUCTION

Constrained optimization

Augmented Lagrangians for general NLPs

Assume your initial problem is **not** a QP (i.e. nonquadratic $c(z)$, nonlinear constraints...).

AL method is still posed as the iteration:

1. minimize the AL function (**HOW?**)

$$\mathcal{L}_\mu(x; y_e, z_e) = c(x) + \frac{1}{2\mu}\|g(x) + \mu y_e\|^2 + \frac{1}{2\mu}\|[h(x) + \mu z_e]_+\|^2$$

2. update multipliers:

$$y^+ = y_e + \tfrac{1}{\mu}g(x^+), \; z^+ = [z_e + \tfrac{1}{\mu}h(x^+)]_+ \tag{16}$$

3. update μ maybe

Constrained trajectory optimization

Problem definition

Constrained trajectory optimization – continuous time

Our objective, in continuous time, is to solve trajectory optimization problem of the form

$$\min_{x,u} \int_0^T \ell(t, x(t), u(t))\, \mathrm{d}t + \ell_T(x(T)) \tag{17a}$$

$$\text{s.t. } \dot{x}(t) = f(t, x(t), u(t)) \tag{17b}$$

$$h(t, x(t), u(t)) \leq 0 \tag{17c}$$

$$h_T(x(T)) \leq 0. \tag{17d}$$

UR10 ballistics video

Quadrotor slalom video

Whole-body MPC on Solo

Constrained trajectory optimization – discrete time

We consider the following discrete-time OCP:

$$
\begin{aligned}
\min_{\boldsymbol{x},\boldsymbol{u}} J(\boldsymbol{x},\boldsymbol{u}) &= \sum_{t=0}^{N-1} \ell_t(x_t,u_t) + \ell_N(x_N) \\
\text{s.t. } x_{t+1} &= f_t(x_t,u_t),\ \ t \in [\![0, N-1]\!] && \longleftrightarrow \lambda_{t+1} \\
x_0 &= \bar{x}_0 && \longleftrightarrow \lambda_0 \\
h_t(x_t,u_t) &\leq 0 && \longleftrightarrow \nu_t \\
h_N(x_N) &\leq 0 && \longleftrightarrow \nu_N
\end{aligned}
\tag{18}
$$

Optimality conditions – the Bellman way

The Bellman principle of optimality The optimal trajectory satisfies the relationship between the cost-to-go functions

$$V_t(x_t) = \min_{u_t} \max_{\nu_t} \ell_t(x_t, u_t) + \nu_t^\top h_t(x_t, u_t) + V_{t+1}(x_{t+1}) \tag{19}$$

where $x_{t+1} = f_t(x_t, u_t)$, and boundary condition

$$V_N(x) = \max_{\nu_N} \ell_N(x) + \nu_N^\top h_N(x). \tag{20}$$

Optimality conditions – the KKT way

The problem Lagrangian is

$$\begin{aligned}\mathscr{L}(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{\lambda}, \boldsymbol{\nu}) = \sum_{t=0}^{N-1} \ell_t(x_t, u_t) + \lambda_{t+1}^\top (f_t(x_t, u_t) - x_{t+1}) + \nu_t^\top h_t(x_t, u_t) \\ + \ell_N(x_N) + \nu_N^\top h_N(x_N) + \lambda_0^\top (x_0 - \bar{x}_0).\end{aligned} \tag{21}$$

We can define the Hamiltonian

$$H_t(x, u, \lambda, \nu) = \ell_t(x, u) + \lambda^\top f_t(x, u) + \nu^\top h_t(x, u) \tag{22}$$

and terminal Lagrangian

$$\mathscr{L}_N(x, \nu) = \ell_N(x) + \nu^\top h_N(x). \tag{23}$$

Thus, the optimality conditions can be written as

$$
\begin{aligned}
\lambda_t &= \nabla_x H_t(x_t, u_t, \lambda_{t+1}, \nu_t) && \text{(24a)}\\
0 &= \nabla_u H_t(x_t, u_t, \lambda_{t+1}, \nu_t) && \text{(24b)}\\
0 &= f_t(x_t, u_t) - x_{t+1} && \text{(24c)}\\
0 &\leq h_t(x_t, u_t) \perp \nu_t \geq 0 && \text{(24d)}\\
0 &\leq h_N(x_N) \perp \nu_N \geq 0 && \text{(24e)}
\end{aligned}
$$

and boundary conditions

$$
\begin{aligned}
x_0 &= \bar{x}_0 && \text{(24f)}\\
\lambda_N &= \nabla_x \mathscr{L}_N(x_N, \nu_N). && \text{(24g)}
\end{aligned}
$$

Can we SQP?

Yes. Start by defining

$$
\begin{aligned}
Q_t &= \nabla^2_{xx} H_t,\ S_t = \nabla^2_{xu} H_t,\ R_t = \nabla^2_{uu} H_t \\
q_t &= \nabla_x H_t,\ r_t = \nabla_u H_t \\
A_t &= \frac{\partial f_t}{\partial x},\ B_t = \frac{\partial f_t}{\partial u},\ s_t = f_t(x_t, u_t) \\
C_t &= \frac{\partial h_t}{\partial x},\ D_t = \frac{\partial h_t}{\partial u},\ d_t = h_t(x_t, u_t)
\end{aligned}
\tag{25}
$$

We can show that the SQP update $(\delta \boldsymbol{x}, \delta \boldsymbol{u}, \boldsymbol{\lambda}^+, \boldsymbol{\nu}^+)$ is obtained by solving the structured QP or *constrained LQR*

$$\min_{\delta \boldsymbol{x}, \delta \boldsymbol{u}} \sum_{t=0}^{N-1} \frac{1}{2} \begin{bmatrix} \delta x_t \\ \delta u_t \end{bmatrix}^\top \begin{bmatrix} Q_t & S_t \\ S_t^\top & R_t \end{bmatrix} \begin{bmatrix} \delta x_t \\ \delta u_t \end{bmatrix} + \ell_{t,x}^\top \delta x_t + \ell_{t,u}^\top \delta u_t \tag{26a}$$

$$\text{s.t. } \delta x_{t+1} = A_t \delta x_t + B_t \delta u_t + \gamma_t \tag{26b}$$

$$C_t \delta x_t + D_t \delta u_t + d_t \leq 0, \tag{26c}$$

$$C_N \delta x_N + d_N \leq 0 \tag{26d}$$

This method is often called iLQR in the literature (Li and Todorov 2004; Giftthaler *et al.* 2018)

- not to be confused with the iLQR of Tassa *et al.* 2012.

(Software) Solutions

- ACADOS (Verschueren *et al.* 2022) implements an SQP-type algorithm, relying on the interior-point method HPIPM for the LQRs (Frison and Diehl 2020).
- CROCODDYL (Mastalli, Budhiraja, *et al.* 2020; Mastalli, Chhatoi, *et al.* 2023) has support for projection-based methods for equality constraints

[2] https://github.com/meco-group/fatrop
[3] https://github.com/machines-in-motion/mim_solvers

(Software) Solutions

- ACADOS (Verschueren *et al.* 2022) implements an SQP-type algorithm, relying on the interior-point method HPIPM for the LQRs (Frison and Diehl 2020).
- CROCODDYL (Mastalli, Budhiraja, *et al.* 2020; Mastalli, Chhatoi, *et al.* 2023) has support for projection-based methods for equality constraints
- FATROP[2] (Vanroye *et al.* 2023) implements an interior-point with an equality-LQR backend
- MIM-SOLVERS[3] (Jordana *et al.* 2023) implements a filter line-search SQP
- our library `aligator`, using proximal/augmented Lagrangian methods based on our prior work (J., Mansard, Carpentier ICRA'22, J., Bambade et *al.* IROS'22 + J., Bambade et *al.* T-RO journal submission)

[2] `https://github.com/meco-group/fatrop`
[3] `https://github.com/machines-in-motion/mim_solvers`

Constrained trajectory optimization

Augmented Lagrangian trajectory optimization with ProxDDP

Note: to simplify presentation, all the h_t are now equality constraints.

The terminal stage value function looks like

$$\begin{aligned} V_N(x) &= \max_{\nu} \ell_N(x) + \nu^\top h_N(x) - \frac{\mu_k}{2}\|\nu - \nu^k\|^2 \\ &= \ell_N(x) + \frac{1}{2\mu_k}\|h_N(x) + \mu_k \nu^k\|^2 - \frac{\mu_k}{2}\|\nu^k\|^2. \end{aligned} \tag{27}$$

Note: to simplify presentation, all the h_t are now equality constraints.

The terminal stage value function looks like

$$\begin{aligned} V_N(x) &= \max_{\nu} \ell_N(x) + \nu^\top h_N(x) - \frac{\mu_k}{2}\|\nu - \nu^k\|^2 \\ &= \ell_N(x) + \frac{1}{2\mu_k}\|h_N(x) + \mu_k \nu^k\|^2 - \frac{\mu_k}{2}\|\nu^k\|^2. \end{aligned} \tag{27}$$

The proximal Bellman recursion is

$$V_t(x) = \min_{u,x'} \max_{\nu,\lambda} \left\{ \mathcal{Q}_t(x,u,\lambda,\nu,x') - \frac{\mu_k}{2}\|\lambda - \underbrace{\lambda^k}_{\text{prox. iteration}}\|^2 - \frac{\mu_k}{2}\|\nu - \nu^k\|^2 \right\} \tag{28}$$

where

$$\mathcal{Q}_t(x,u,\lambda,\nu,x') \stackrel{\text{def}}{=} \ell_t(x,u) + \nu^\top h_t(x,u) + \lambda^\top (f_t(x,u) - x') + V_{t+1}(x'). \tag{29}$$

General principle. Solve recursion DDP/iLQR-style with a quadratic model!

General principle. Solve recursion DDP/iLQR-style with a quadratic model!

Recursion hypothesis. We posit that the next-step value function variation is

$$\delta V_{t+1}(\delta x) \approx p_{t+1}^\top \delta x + \frac{1}{2} \delta x^\top P_{t+1} \delta x. \tag{30}$$

Goal. Close the recursion, by using Bellman's equation!

Then, solve for $(\delta u_t, \delta \nu_t, \delta \lambda_{t+1}, \delta x_{t+1})$ as functions of δx_t as follows:

$$\underbrace{\begin{bmatrix} R_t & D_t^\top & B_t^\top & \\ D_t & -\mu_k I & & \\ B_t & & -\mu_k I & -I \\ & & -I & P_{t+1} \end{bmatrix}}_{\stackrel{\text{def}}{=}\mathcal{K}_t} \begin{bmatrix} \delta u_t \\ \delta \nu_t \\ \delta \lambda_{t+1} \\ \delta x_{t+1} \end{bmatrix} = - \begin{bmatrix} r_t + S_t^\top \delta x_t \\ \bar{d}_t^k - \mu_k \nu_t + C_t \delta x_t \\ \bar{s}_t^k - \mu_k \lambda_{t+1} + A_t \delta x_t \\ p_{t+1} \end{bmatrix} \tag{31}$$

where, $\bar{d}_t^k = d_t + \mu_k \nu_t^k$, $\bar{s}_t^k = s_t + \mu_k \lambda_{t+1}^k$.

Then, solve for $(\delta u_t, \delta\nu_t, \delta\lambda_{t+1}, \delta x_{t+1})$ as functions of δx_t as follows:

$$\underbrace{\begin{bmatrix} R_t & D_t^\top & B_t^\top & \\ D_t & -\mu_k I & & \\ B_t & & -\mu_k I & -I \\ & & -I & P_{t+1} \end{bmatrix}}_{\stackrel{\text{def}}{=}\mathcal{K}_t} \begin{bmatrix} \delta u_t \\ \delta\nu_t \\ \delta\lambda_{t+1} \\ \delta x_{t+1} \end{bmatrix} = - \begin{bmatrix} r_t + S_t^\top \delta x_t \\ \bar{d}_t^k - \mu_k \nu_t + C_t \delta x_t \\ \bar{s}_t^k - \mu_k \lambda_{t+1} + A_t \delta x_t \\ p_{t+1} \end{bmatrix} \tag{31}$$

where, $\bar{d}_t^k = d_t + \mu_k \nu_t^k$, $\bar{s}_t^k = s_t + \mu_k \lambda_{t+1}^k$. As δx_t is unknown, we can (in DDP fashion) extract a *parametric* solution in feedforward/feedback form:

$$\begin{bmatrix} k_t & K_t \\ \zeta_t & Z_t \\ \xi_{t+1} & \Xi_{t+1} \\ m_t & M_t \end{bmatrix} = -\mathcal{K}_t^{-1} \begin{bmatrix} r_t & S_t^\top \\ \bar{d}_t^k - \mu_k \nu_t & C_t \\ \bar{s}_t^k - \mu_k \lambda_{t+1} & A_t \\ p_{t+1} & 0 \end{bmatrix} \tag{32}$$

The value function model update is given by

$$P_t = Q_t + S_t K_t + C_t^\top Z_t + B_t^\top \Xi_{t+1} \tag{33a}$$

$$p_t = q_t + S_t k_t + C_t^\top \zeta_t + B_t^\top \xi_{t+1} \tag{33b}$$

such that $\delta V_t(\delta x) \approx p_t^\top \delta x + \frac{1}{2}\delta x^\top P_t \delta x$.

Thereby closing the recursion.

Initial stage. The initial stage constraint is $\bar{x}_0 - x_0 = 0$.

The update $(\delta x_0, \delta\lambda_0)$ satisfies

$$\begin{bmatrix} P_0 & -I \\ -I & -\mu_k I \end{bmatrix} \begin{bmatrix} \delta x_0 \\ \delta\lambda_0 \end{bmatrix} = - \begin{bmatrix} p_0 \\ \bar{x}_0 \end{bmatrix} \tag{34}$$

This leaves the way open to some **extensions**, e.g. initial constraints $g_0(x_0) = 0$.

Linear vs. nonlinear rollouts

Once $(\delta x_0, \delta\lambda_0)$ is computed, we can reconstruct the update for the trajectory:

Linear rollout (a.k.a. SQP)

$$\delta u_t = k_t + K_t \delta x_t \tag{35a}$$

$$\delta \nu_t = \zeta_t + Z_t \delta x_t \tag{35b}$$

$$\delta \lambda_{t+1} = \xi_{t+1} + \Xi_{t+1} \delta x_t \tag{35c}$$

$$\delta x_{t+1} = m_t + M_t \delta x_t \tag{35d}$$

Nonlinear rollout (DDP-style)

$$u_t^+ = u_t + k_t + K_t \delta x_t \tag{36a}$$

$$\nu_t^+ = \nu_t + \zeta_t + Z_t \delta x_t \tag{36b}$$

$$\lambda_{t+1}^+ = \lambda_{t+1} + \xi_{t+1} + \Xi_{t+1} \delta x_t \tag{36c}$$

$$x_{t+1}^+ = f_t(x_t^+, u_t^+) - \mu_k \lambda_{t+1}^+ \tag{36d}$$

$$\delta x_{t+1} = x_{t+1}^+ - x_{t+1} \tag{36e}$$

Read the preprint!

ProxDDP: Proximal Constrained Trajectory Optimization

Wilson Jallet[*,1,2], Antoine Bambade[1], Etienne Arlaud[1], Sarah El-Kazdadi[1], Nicolas Mansard[2] and Justin Carpentier[1]

Abstract*—Trajectory optimization (TO) has proven, over the last decade, to be a versatile and effective framework for robot control. Several numerical solvers have been demonstrated to be fast enough to allow recomputing full-dynamics trajectories for various systems at control time, enabling model predictive control (MPC) of complex robots. These first implementations of MPC in robotics predominantly utilize some differential dynamic programming (DDP) variant for its computational speed and ease of use in constraint-free settings. Nevertheless, many scenarios in robotics call for adding hard constraints in TO problems (e.g., torque limits, obstacle avoidance), which existing solvers, based on DDP, often struggle to handle. Effectively addressing path constraints still poses optimization challenges (e.g., numerical stability, efficiency, accuracy of constraint satisfaction) that we propose to solve by combining advances in numerical optimization with the foundational efficiency of DDP. In this article, we leverage proximal methods for constrained optimization and introduce a DDP-like method to achieve fast, constrained trajectory optimization with an efficient warm-starting strategy particularly suited for MPC applications. Compared to earlier solvers, our approach effectively manages hard constraints without warm-start limitations and exhibits commendable convergence accuracy. Additionally, we leverage the computational efficiency of DDP, enabling real-time resolution of complex problems such as whole-body quadruped locomotion. We provide a complete implementation as part of an open-source and flexible C++ trajectory optimization library called Aligator. These algorithmic contributions are validated through several trajectory planning scenarios from the robotics literature and the real-time whole-body MPC of a quadruped robot.

***Index Terms*—Optimization and Optimal Control, Legged Robots, Model-Predictive Control**

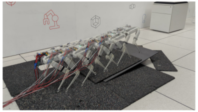


Fig. 1. **Solo-12 walking on an *unmodelled* slope** using the whole-body MPC framework based on primal-dual augmented Lagrangian techniques.

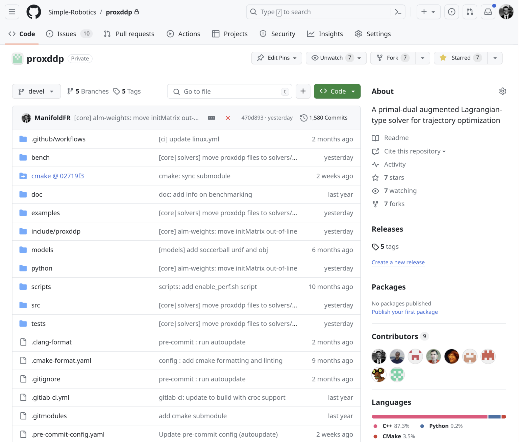
I. Introduction

Trajectory optimization is an efficient and generic approach for controlling complex dynamical systems such as robots. It is a principled framework for describing desired behaviors and generating motion. A workhorse in modern robotics, it has become a crucial ingredient in both kinodynamic planning and model predictive control (MPC) over the past decade, enabled by the increasing performance of computer chips and algorithmic enhancements alleviating previous computational bottlenecks. Notably, recent progress in both software and hardware has enabled the real-time computation of numerical quantities commonly involved in

Optimal control problems (OCPs) are, by nature, infinite-dimensional optimization problems, which are largely not solvable in closed form. However, they can be solved numerically. On the one hand, there are *indirect* methods for OCPs, based on deriving their optimality conditions [4]. On the other hand, there are *direct* methods [5] which transcribe OCP problems into nonlinear programs (NLPs) of finite dimensions.

Direct methods, whichever the method of transcription, attempt resolution by utilizing a nonlinear programming approach, either leveraging general-purpose and off-the-shelf solvers such as Ipopt [6] or Snopt [7], or a more tailored solution. Several approaches are considered in the literature to solve them in practice. We will argue why, in the robotics community, differential dynamic programming-based solvers are seen as a promising research direction, notably for deploying receding horizon control schemes for real-time robot control. One transcription method is *collocation*, which approximates the OC problem using a finite-dimensional basis functions such as polynomials. Another transcription method is *shooting methods*. They use a discretization of the system dynamics through numerical integration, which is generic, efficient, and easy to implement. For these shooting methods, there exist structure-exploiting solvers leveraging Riccati recursion [8]–[11], such as the differential dynamic programming (DDP) [12] algorithm. DDP is one of the earliest such methods and a reference in nonlinear trajectory optimization, known to have quadratic convergence [13], and has several variants such as the iterative

Library to be publicly released **soon** (multi-team effort, please contribute!).



AGIMUS
INNOVATIVE ROBOTICS FOR AGILE PRODUCTION

[1] R. T. Rockafellar, *Convex Analysis*. Princeton University Press, Jan. 12, 1997, 482 pp., ISBN: 978-0-691-01586-6. Google Books: 1TiOka9bx3sC.

[2] P. Gill, W. Murray, and M. Saunders, **“Snopt: An sqp algorithm for large-scale constrained optimization,”** *SIAM Journal on Optimization*, vol. 12, pp. 979–1006, Apr. 26, 2002. DOI: 10.2307/20453604.

[3] A. Wächter and L. T. Biegler, **“On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming,”** *Mathematical Programming*, vol. 106, no. 1, pp. 25–57, Mar. 2006, ISSN: 0025-5610, 1436-4646. DOI: 10/c6j59p. [Online]. Available: http://link.springer.com/10.1007/s10107-004-0559-y (visited on 11/12/2021).

[4] A. R. Conn, N. I. M. Gould, and P. L. Toint, *Lancelot: A Fortran Package for Large-Scale Nonlinear Optimization (Release A)*, 1st ed. Springer Publishing Company, Incorporated, Nov. 2010, 330 pp., ISBN: 978-3-642-08139-2.

[5] J. Nocedal and S. J. Wright, *Numerical Optimization* (Springer Series in Operations Research), 2nd ed. New York: Springer, 2006, 664 pp., ISBN: 978-0-387-30303-1.

[6] D. Goldfarb and A. Idnani, **"A numerically stable dual method for solving strictly convex quadratic programs,"** *Mathematical Programming*, vol. 27, no. 1, pp. 1–33, Sep. 1, 1983, ISSN: 1436-4646. DOI: 10.1007/BF02591962. [Online]. Available: https://doi.org/10.1007/BF02591962 (visited on 12/06/2023).

[7] A. Conn, N. Gould, and P. Toint, **"A globally convergent augmented lagrangian algorithm for optimization with general constraints and simple bounds,"** *SIAM Journal on Numerical Analysis*, vol. 28, Apr. 1, 1991. DOI: 10.1137/0728030.

[8] J. Carpentier, R. Budhiraja, and N. Mansard, **"Proximal and sparse resolution of constrained dynamic equations,"** in *Robotics: Science and Systems XVII*, Robotics: Science and Systems Foundation, Jul. 12, 2021, ISBN: 978-0-9923747-7-8. DOI: 10.15607/RSS.2021.XVII.017. [Online]. Available: http://www.roboticsproceedings.org/rss17/p017.pdf (visited on 11/10/2022).

[9] B. Stellato, G. Banjac, P. Goulart, A. Bemporad, and S. Boyd, **"Osqp: An operator splitting solver for quadratic programs,"** *Mathematical Programming Computation*, vol. 12, no. 4, pp. 637–672, Dec. 2020, ISSN: 1867-2949, 1867-2957. DOI: 10.1007/s12532-020-00179-2. [Online]. Available: http://link.springer.com/10.1007/s12532-020-00179-2 (visited on 01/23/2021).

[10] H. J. Ferreau, C. Kirches, A. Potschka, H. G. Bock, and M. Diehl, **"Qpoases: A parametric active-set algorithm for quadratic programming,"** *Mathematical Programming Computation*, vol. 6, no. 4, pp. 327–363, Dec. 1, 2014, ISSN: 1867-2957. DOI: 10.1007/s12532-014-0071-1. [Online]. Available: https://doi.org/10.1007/s12532-014-0071-1 (visited on 12/06/2023).

[11] B. Hermans, A. Themelis, and P. Patrinos, **"Qpalm: A newton-type proximal augmented lagrangian method for quadratic programs,"** *2019 IEEE 58th Conference on Decision and Control (CDC)*, pp. 4325–4330, Dec. 2019. DOI: 10.1109/CDC40024.2019.9030211. arXiv: 1911.02934. [Online]. Available: http://arxiv.org/abs/1911.02934 (visited on 04/01/2021).

[12] A. De Marchi, **"On a primal-dual newton proximal method for convex quadratic programs,"** *Computational Optimization and Applications*, vol. 81, no. 2, pp. 369–395, Mar. 2022, ISSN: 0926-6003, 1573-2894. DOI: 10.1007/s10589-021-00342-y. [Online]. Available: https://link.springer.com/10.1007/s10589-021-00342-y (visited on 02/18/2022).

[13] A. Bambade, F. Schramm, S. E. Kazdadi, S. Caron, A. Taylor, and J. Carpentier, **Proxqp: An efficient and versatile quadratic programming solver for real-time robotics applications and beyond,** Sep. 1, 2023. [Online]. Available: https://inria.hal.science/hal-04198663 (visited on 12/06/2023).

[14] R. T. Rockafellar, **“Augmented lagrangians and applications of the proximal point algorithm in convex programming,”** *Mathematics of Operations Research*, vol. 1, no. 2, pp. 97–116, 1976, ISSN: 0364-765X. JSTOR: 3689277. [Online]. Available: https://www.jstor.org/stable/3689277 (visited on 03/18/2021).

[15] W. Li and E. Todorov, **“Iterative linear quadratic regulator design for nonlinear biological movement systems,”** in *Proceedings of the First International Conference on Informatics in Control, Automation and Robotics*, Setúbal, Portugal: SciTePress - Science, 2004, pp. 222–229, ISBN: 978-972-8865-12-2. DOI: 10/fn7nnp. [Online]. Available: http://www.scitepress.org/DigitalLibrary/Link.aspx?doi=10.5220/0001143902220229 (visited on 01/07/2021).

[16] M. Giftthaler, M. Neunert, M. Stäuble, J. Buchli, and M. Diehl, **"A family of iterative gauss-newton shooting methods for nonlinear optimal control,"** *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018. DOI: 10.1109/IROS.2018.8593840.

[17] Y. Tassa, T. Erez, and E. Todorov, **"Synthesis and stabilization of complex behaviors through online trajectory optimization,"** in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems.*, Oct. 1, 2012, pp. 4906–4913, ISBN: 978-1-4673-1737-5. DOI: 10.1109/IROS.2012.6386025.

[18] R. Verschueren *et al.*, **"Acados—a modular open-source framework for fast embedded optimal control,"** *Mathematical Programming Computation*, vol. 14, no. 1, pp. 147–183, Mar. 1, 2022, ISSN: 1867-2957. DOI: 10.1007/s12532-021-00208-8. [Online]. Available: https://doi.org/10.1007/s12532-021-00208-8 (visited on 07/03/2023).

AGIMUS
INNOVATIVE ROBOTICS FOR AGILE PRODUCTION

[19] G. Frison and M. Diehl, **"Hpipm: A high-performance quadratic programming framework for model predictive control,"** *IFAC-PapersOnLine*, 21st IFAC World Congress, vol. 53, no. 2, pp. 6563–6569, Jan. 1, 2020, ISSN: 2405-8963. DOI: 10.1016/j.ifacol.2020.12.073. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2405896320303293 (visited on 07/03/2023).

[20] C. Mastalli, R. Budhiraja, *et al.*, **"Crocoddyl: An efficient and versatile framework for multi-contact optimal control,"** *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2536–2542, May 2020. DOI: 10.1109/ICRA40945.2020.9196673. [Online]. Available: https://ieeexplore.ieee.org/document/9196673/ (visited on 07/02/2023).

[21] C. Mastalli, S. P. Chhatoi, T. Corbéres, S. Tonneau, and S. Vijayakumar, **"Inverse-dynamics mpc via nullspace resolution,"** *IEEE Transactions on Robotics*, pp. 1–20, 2023, ISSN: 1941-0468. DOI: 10.1109/TRO.2023.3262186.

AGIMUS
INNOVATIVE ROBOTICS FOR AGILE PRODUCTION

[22] L. Vanroye, A. Sathya, J. De Schutter, and W. Decré, **"Fatrop : A fast constrained optimal control problem solver for robot trajectory optimization and control,"** arXiv: `2303.16746 [cs, math]`. (Mar. 29, 2023), [Online]. Available: `http://arxiv.org/abs/2303.16746` (visited on 07/02/2023), preprint.

[23] A. Jordana, S. Kleff, A. Meduri, J. Carpentier, N. Mansard, and L. Righetti, **"Stagewise implementations of sequential quadratic programming for model-predictive control,"** (Dec. 2023), preprint.

AGIMUS
INNOVATIVE ROBOTICS FOR AGILE PRODUCTION