UNIVERSIDAD TECNICA
FEDERICO SANTA MARIA

Departamento de Electrónica

CARACTERIZACIÓN DE TIEMPO DE EJECUCIÓN DE SOLVER OSQP PARA APLICACIONES EN CONTROL PREDICTIVO POR MODELO

MANUEL SIMON VILLACORTA

MEMORIA DE TITULACIÓN PARA OPTAR AL TÍTULO DE INGENIERO CIVIL ELECTRÓNICO

PROFESOR SUPERVISOR	:	GONZALO CARVAJAL BARRERA
PROFESOR CORREFERENTE	:	JUAN CARLOS AGÜERO VÁSQUEZ
PROFESOR CORREFERENTE	:	CÉSAR SILVA JIMÉNEZ

10 DE DICIEMBRE DE 2025

CONSTANCIA DE VALIDACIÓN Y CONFIDENCIALIDAD DE MONOGRAFÍA A REPOSITORIO ACADÉMICO

1.- IDENTIFICACIÓN DEL TRABAJO ACADÉMICO

Tipo de monografía (marcar una opción): ☒ Memoria o trabajo de título ☐ Tesis de Postgrado

Título del trabajo: Caracterización de tiempo de ejecución de solver OSQP para aplicaciones en control predictivo por modelo.

Nombre del candidato(a): Manuel Simon Villacorta.

Carrera / Grado: Ingeniería Civil Electrónica.

Campus: Casa Central Valparaíso. **Departamento:** Electrónica.

2.- VALIDACIÓN DEL PROFESOR GUÍA/DIRECTOR DE TESIS

Yo, Gonzalo Carvajal Barrera , en mi calidad de profesor(a) guía/director(a) del trabajo académico mencionado anteriormente **DEJO CONSTANCIA** que:

- He revisado esta versión del documento y corresponde a la versión final aprobada del trabajo.
- El trabajo cumple con los requisitos académicos y de formato establecidos por la institución.

3.- EVALUACIÓN DE CONFIDENCIALIDAD POR PROPIEDAD INDUSTRIAL (marcar una opción)

☒ El trabajo **NO contiene** información que amerite confidencialidad y puede ser publicado de inmediato en repositorio con acceso abierto.

☐ El trabajo **CONTIENE** información con potenciales implicancias de propiedad industrial o intelectual y requiere un periodo de confidencialidad (**embargo**) por **(marcar una opción)**:

☐6 meses ☐ 12 meses ☐ 2 años ☐ 3 años ☐ 5 años ☐ 10 años

Fundamentación de la necesidad de confidencialidad (obligatorio si se solicita embargo):

4.- FIRMAS

Profesor(a) guía o director(a) de memoria o tesis:

Fecha: 30/12/2025 **Firma:** _____

Estudiante o Candidato(a):

Fecha: 30/12/2025 **Firma:** _____

Este formulario debe ser insertado como página 2 de la memoria o tesis, completado y firmado por estudiante y profesor(a) antes de la entrega en portal PRISMA de Biblioteca USM.

Resumen

El presente trabajo expone los resultados de un estudio sobre los tiempos de ejecución del solver Operator Splitting Quadratic Program (`OSQP`), una herramienta de optimización convexa ampliamente utilizada en aplicaciones de Model Predictive Control (MPC). El análisis se orienta a evaluar su viabilidad en sistemas embebidos, entornos caracterizados por recursos computacionales limitados. A partir de un código desarrollado en MATLAB, se generaron instancias aleatorias de problemas con distintas dimensiones, sobre las cuales se evaluó el desempeño del solver bajo diversas configuraciones de parámetros. El objetivo principal es caracterizar su comportamiento computacional, analizando su precisión, escalabilidad y estabilidad temporal, con miras a una eventual implementación en plataformas embebidas de bajo costo.

Como parte del proceso de validación, se realizaron experimentos comparativos con el solver nativo de MATLAB, con el propósito de verificar la equivalencia funcional del algoritmo y garantizar la consistencia de los resultados obtenidos. Además, se llevaron a cabo pruebas experimentales en dos tarjetas Raspberry Pi, lo que permitió evaluar el rendimiento del solver en un entorno embebido real, considerando tanto sus limitaciones de hardware como sus capacidades para operar en tiempo real. Finalmente, se incluyeron casos de estudio asociados a Sistemas de Energía Distribuida (DER), con el fin de analizar el comportamiento de `OSQP` en aplicaciones prácticas de MPC.

Abstract

This work presents the results of a study on the execution times of the Operator Splitting Quadratic Program (OSQP) solver, a convex optimization tool widely used in Model Predictive Control (MPC). The analysis focuses on assessing its feasibility for embedded systems, which are typically characterized by limited computational resources.

Using a MATLAB based implementation, random instances of problems with varying dimensions were generated to evaluate the solver's performance under different parameter configurations. The main objective is to characterize its computational behavior, examining accuracy, scalability, and timing stability with a view toward potential deployment on low cost embedded platforms. As part of the validation process, comparative experiments were performed against MATLAB's native solver to verify the functional equivalence of the algorithm and ensure the consistency of the obtained results. In addition, experimental tests were conducted on two Raspberry Pi boards, enabling evaluation of the solver's performance in a real embedded environment, considering both hardware constraints and real-time operation requirements. Finally, case studies involving Distributed Energy Resources (DER) were included to analyze OSQP's behavior in practical MPC applications.

Para el desarrollo del trabajo desarrollado en esta memoria de título se recibió apoyo de los proyectos internos USM PI_M_23_05 y PI_LIR_25_12, y también del proyecto ANID-Basal Project Grant AFB240002 (AC3E).

Índice general

Índice de tablas

Índice de figuras

1 Introducción

El presente trabajo de título tiene como objetivo evaluar el desempeño del solver `OSQP` [1] en la resolución de problemas de Quadratic Programming (QP) típicamente encontrados en algoritmos de Control Predictivo Basado en Modelos (MPC, por sus siglas en inglés). El estudio acá reportado se enmarca en una línea de investigación desarrollada en los últimos años en el Departamento de Electrónica, orientada al diseño, análisis e implementación de algoritmos MPC para sistemas prácticos que requirean operar sujetos a restricciones de tiempo real y baja latencia.

La implementación de lazos MPC en sistemas lineales sujetos a restricciones lineales en estados y salidas requieren resolver en línea un problema de optimizacion cuadratica para determinar la actuación óptima en cada intervalo de muestreo del controlador. En entornos prácticos, la resolución del problema QP suele representar el principal cuello de botella en términos de tiempo de ejecución del lazo. Hoy en día, la creciente adopción de MPC en entornos basados en sistemas embebidos ha impulsado la necesidad de contar con solvers de optimización que combinen precisión numérica y eficiencia computacional para ser implementados en plataformas con recursos limitados.

A partir de un estudio exploratorio previo desarrollado como parte de una memoria de título [2], se identificó al solver `OSQP` como una opción especialmente prometedora para su implementación en sistemas embebidos con requerimientos de baja latencia. Entre varias alternativas evaluadas, `OSQP` se diferenció por contar con una buena documentación y ejemplos de uso, desarrollo activo y repositorios de código actualizados, contar con código optimizado en C, integración con MATLAB, y reportes de benchmarks que indicaban tiempos de resolución competitivos comparados con otras alternativas populares. No obstante, dicho estudio se centró principalmente en evaluar aspectos funcionales y de usabilidad, por lo que se utilizaron configuraciones por defecto, sin profundizar en los detalles de configuración y evaluar el impacto que la configuración de parámetros tiene sobre el rendimiento del solver en problemas de control específicos.

En este contexto, el trabajo desarrollado en esta memoria apunta a extender la caracterización del solver `OSQP`, presentando un análisis sistemático del desempeño del solver, explorando cómo diferentes configuraciones afectan el tiempo de ejecución y la precisión de las soluciones en diversos escenarios. Los resultados buscan aportar criterios prácticos que orienten futuras implementaciones de MPC eficientes y adaptadas a requisitos de tiempo real.

Este capítulo introduce el contexto y motivación del trabajo, presenta el planteamiento del problema junto con los objetivos, el alcance y las principales contribuciones de la memoria. Asimismo, se justifica la necesidad de realizar una caracterización exhaustiva del solver `OSQP` y

se ofrece una visión general de la organización del informe.

1.1. Contexto y motivación

MPC es una técnica de control avanzada basada en optimización, que utiliza un modelo matemático del sistema junto con una medición o estimación de sus estados en un instante determinado. A partir de esta información, el controlador puede predecir el comportamiento futuro del sistema y calcular los valores óptimos de las entradas de control para que la salida siga una referencia deseada a lo largo un horizonte de muestras futuras, asegurando al mismo tiempo el cumplimiento de restricciones en las entradas y los estados del sistema [3].

Desde el punto de vista computacional, las implementaciones de MPC para sistemas lineales implican resolver, en cada instante de muestreo, un problema de optimización que se formula como un problema QP. Este proceso busca encontrar la secuencia óptima de valores de entrada que minimicen una función de costo (por ejemplo, el error respecto a una referencia) definida para un horizonte de predicción de N muestras futuras. A diferencia de métodos de control clásicos como el control proporcional, integral y derivativo (PID) o el Linear Quadratic Regulator (LQR), MPC permite considerar explícitamente restricciones en las entradas y estados, lo cual otorga una ventaja significativa en sistemas con limitaciones físicas que deban cumplirse por costo, seguridad o normativas (como límites de tensión, corriente o velocidad).

Actualmente, existen distintos solvers para resolver problemas QP que se adaptan a entornos de operación específicos, considerando la forma de resolver el problema de optimización, la facilidad de uso, la integración con otras plataformas, etc. En cuanto al licenciamiento, algunos son de código abierto, mientras que otros requieren licencias de pago. Como alternativas a `OSQP` se pueden mencionar las siguientes:

- `quadprog` [4]: solver integrado en MATLAB que emplea algoritmos como el método de punto interior y el método activo. Su estrecha integración con dicho entorno lo hace especialmente conveniente para su uso en la implementación de lazos de control en Simulink. Sin embargo, esta dependencia del ecosistema MATLAB restringe su despliegue en sistemas embebidos que podrían no disponer del hardware ni del sistema operativo necesarios para ejecutarlo.

- `qpOASES` [5]: este solver emplea un método diseñado para resolver secuencias de problemas QP con estructuras similares, lo que lo hace especialmente eficiente en aplicaciones de MPC, donde los problemas de optimización se repiten con ligeras variaciones en cada instante de control. Además, la implementación es de código abierto, lo cual ofrece ventajas en términos de transparencia y adaptabilidad. No obstante, al momento de iniciar este trabajo, se detectó que el desarrollo de este solver se encuentra inactivo desde hace varios años, por lo que no se considera una alternativa adecuada para este estudio.

- `Gurobi` [6] y `MOSEK` [7] son solvers de licencia de pago orientados a entornos industriales y de producción, ampliamente utilizados en áreas como optimización financiera, logística y gestión de recursos, donde normalmente se requiere resolver problemas de gran escala. Si bien también se emplean en aplicaciones de ingeniería y control, su carácter propietario, los costos asociados a la adquisición y mantención de licencias, y las restricciones de uso en ciertos entornos computacionales limitan su adopción en proyectos académicos o de investigación.

Para el desarrollo de este trabajo y en el ámbito académico resultan especialmente atractivas las alternativas de software no comercial. En esta línea, el trabajo de título reportado en [2] exploró distintos solvers de código abierto y destacó `OSQP` por su implementación eficiente en C, su orientación a problemas de gran escala con bajo consumo de recursos y su uso documentado en aplicaciones de MPC [1, 8]. No obstante, dicho estudio se centró en validar la funcionalidad del controlador y en comparar tiempos de ejecución globales entre solvers, sin realizar un análisis sistemático del comportamiento interno de `OSQP` ni del impacto de sus parámetros de configuración sobre el tiempo de cómputo y la calidad de las soluciones. Como consecuencia, el trabajo deja abierta la posibilidad de una sintonización más fina orientada a optimizar los tiempos de ejecución.

Considerando que `OSQP` permite un ajuste fino de sus parámetros internos, cuenta con mantenimiento activo y actualizaciones periódicas, su portabilidad hacia distintas plataformas de cómputo se ve considerablemente facilitada. Estas características permiten adaptarlo a una amplia variedad de escenarios, incluidos sistemas embebidos, manteniendo control sobre aspectos como precisión, velocidad de convergencia y carga computacional. Por estas razones, `OSQP` se posiciona como una alternativa robusta, flexible y plenamente coherente con los objetivos de esta investigación.

1.2. Planteamiento del problema

El presente trabajo tiene como objetivo realizar una exploración sistemática de los parámetros de usuario del solver `OSQP`, evaluando el efecto que estos tienen sobre el tiempo de ejecución y la precisión de las soluciones obtenidas en la resolución de problemas QP con distintas dimensiones.

Para abordar este objetivo, el desarrollo del trabajo se estructura en las siguientes etapas:

- **Validación funcional de `OSQP` en la resolución de problemas QP de distinta dimensión:** Esto incluye la generación de instancias de problemas QP, su resolución con `OSQP` y `quadprog`, y la comparación de las soluciones en términos de valor de la solución primal y tiempos de ejecución. `quadprog` se utiliza como *baseline* por ser un solver ampliamente validado e integrado en MATLAB, lo que brinda un referente confiable en factibilidad de solución y precisión de esta. De esta forma, discrepancias con `OSQP` encontradas en el análisis posterior pueden atribuirse al algoritmo empleado por el solver y no a errores de implementación.

- **Traducción de los problemas a lenguaje C:** Permite una caracterización más precisa del tiempo de ejecución en un entorno compilado, sin los *overheads* del intérprete de MATLAB. Además, constituye el primer paso para portar y ejecutar las instancias en microcontroladores y microprocesadores de distintas gamas y arquitecturas.

- **Exploración de parámetros y evaluación en distintas plataformas:** Se realiza un estudio sobre cómo afecta la modificación de los parámetros propios del solver `OSQP` al tiempo de ejecución y a la precisión de las soluciones. En primera instancia, las pruebas se ejecutan en un computador portátil, y posteriormente se extienden algunas configuraciones seleccionadas para su evaluación en plataformas Raspberry Pi. A partir de esta evaluación, se derivan directrices genéricas que faciliten la sintonización de parámetros para aplicaciones de MPC que requieran implementación en plataformas

embebidas, considerando balance entre precisión de la solución y tiempo de ejecución.

- **Validación de los hallazgos en aplicaciones representativas de lazo de control:** Para validar la utilidad de las directrices derivadas del paso anterior, se evalúan dos casos de estudio de lazos de control MPC reportados en trabajos de título recientes [9, 10], originalmente descritos en MATLAB/Simulink, con el objetivo de verificar que las recomendaciones propuestas se mantienen efectivas al ser aplicadas en escenarios de control representativos.
- **Desarrollo de repositorio documental:** Se crea un repositorio estructurado que contiene los scripts, códigos fuente y configuraciones utilizadas durante el estudio, con el fin de asegurar la trazabilidad, reproducibilidad y respaldo del trabajo realizado en cada una de las etapas [11].

1.3. Alcances y contribuciones

Este documento se centra en el estudio y caracterización de los tiempos de ejecución del solver `OSQP` bajo distintas configuraciones, en el contexto de problemas QP. A continuación, se detallan los principales alcances considerados:

- Para la evaluación del tiempo de ejecucion se considera un computador portátil y placas Raspberry Pi de distintas gamas. Estas plataformas se eligen por su accesibilidad, amplia disponibilidad y por ejecutar un sistema operativo, lo que facilita la implementación y uso del solver. Aunque la literatura reporta que `OSQP` puede ejecutarse en microcontroladores de menores prestaciones, esos dispositivos se excluyen en este estudio por restricciones de tiempo y se consideran como trabajo futuro.
- El solver `OSQP` será ejecutado empleando un único núcleo de procesamiento. Aunque admite paralelización mediante bibliotecas especializadas, este estudio se restringe a la ejecución secuencial para facilitar la comparación.
- Con el objetivo de evaluar la utilidad práctica del solver dentro de un lazo de control real, se seleccionaron casos de estudio representativos del área de la electrónica de potencia, previamente validados y documentados. En este trabajo, el análisis se centra en verificar la equivalencia funcional del solver en dos escenarios específicos. En el primero, se compara la resolución de problemas QP utilizando `OSQP` en MATLAB con la resolución obtenida mediante la implementación en C del solver, configurando en ambos casos los parámetros de acuerdo con las directrices derivadas de este estudio. En el segundo escenario, se sustituye el solver `quadprog` por `OSQP` dentro del entorno MATLAB, ajustando igualmente sus parámetros con el fin de analizar su desempeño.

La principal contribución de este trabajo es la caracterización del desempeño de `OSQP` bajo distintas configuraciones y su evaluación en sistemas embebidos, con evidencia empírica de sus capacidades y limitaciones. Este análisis permite juzgar la viabilidad de `OSQP` en control en tiempo real, considerando las restricciones computacionales típicas de plataformas embebidas.

Complementariamente, aunque existen propuestas con aceleradores de hardware, por ejemplo implementaciones de MPC en FPGA [12], una evaluación justa de esas alternativas exige primero conocer el máximo desempeño alcanzable con soluciones puramente de software. En particular, es clave establecer el potencial de `OSQP` en configuraciones óptimas para decidir

si vale la pena migrar a plataformas de mayor complejidad y costo, o si basta con una implementación eficiente sobre CPU embebida. Los resultados buscan servir como línea base de desempeño mínimo esperado para la implementación de un lazo MPC en hardware de bajo consumo y como referencia para comparaciones y mejoras futuras.

1.4. Organización del informe

El enfoque principal de este trabajo es de carácter experimental, y sus principales entregables incluyen códigos y documentación técnica diseñada para su ejecución. Estos códigos, junto con detalles técnicos, están disponibles en un repositorio [11]destinado a su validación y posterior reproducción.

El presente informe complementa estos entregables proporcionando información adicional y destacando resultados importantes obtenidos durante la exploración.

La estructura de los capítulos se organiza de la siguiente manera:

- **Capítulo 2:** Expone aspectos teóricos de MPC y presenta los solvers `quadprog` y `OSQP`.
- **Capítulo 3:** Detalla la metodología empleada, las métricas de evaluación utilizadas, una comparación preliminar entre `quadprog` y `OSQP`. Además, se llevan a cabo pruebas modificando los parámetros de usuario de `OSQP`.
- **Capítulo 4:** Continúa la exploración del desempeño de `OSQP`, esta vez eligiendo pruebas más específicas y evaluándolo en entornos embebidos. Además se evalua el desempeño del solver `OSQP` en diferentes escenarios, dos se basan en sistemas DER, donde se analizan tiempos de ejecución y equivalencia funcional entre solvers en lazo de control, y uno compara el rendimiento de `OSQP` en CPU con una implementación de aceleradores en FPGA, comparando sus tiempos de ejecución en un problema de dimensiones similares.
- **Capítulo 5:** Resume las principales conclusiones y aportes derivados de este trabajo, además de sugerir posibles trabajos futuros.

2 | Antecedentes

En este capítulo se presentan y discuten los fundamentos teóricos del MPC, junto con una introducción a los solvers `quadprog` y `OSQP`. Los contenidos teóricos se incluyen para contextualizar el *script* utilizado en la generación de problemas de MPC en este trabajo, y si bien no son indispensables para seguir el desarrollo, se incorporan por completitud y como referencia para quienes deseen modificar dicho *script*. Parte de la formulación teórica corresponde a extractos de un trabajo de Andrew Morrison, Ángel Cedeño y Gonzalo Carvajal, incluidos con permiso de los autores y con ajustes menores de notación para uniformar el uso de variables con el resto de este documento. El solver `quadprog` se describe delimitando la clase de problemas QP que resuelve, a fin de facilitar su comparación con `OSQP`. En contraste, `OSQP` se presenta con mayor detalle: se especifica la forma del problema QP que aborda, el algoritmo empleado y los parámetros del solver junto con sus valores por defecto.

2.1. Fundamentos de MPC

Dadas las condiciones de que el sistema a controlar es lineal, se define una función de costo cuadrática, y las restricciones sobre las entradas y estados del sistema son lineales (por ejemplo, restricciones de caja), entonces se dice que el problema de MPC es lineal. Dado un problema MPC lineal, el problema de optimización a resolver en cada instante de muestreo se puede representar como la formulación canónica de un problema de QP con restricciones de igualdad y desigualdad, la cual tiene la forma:

$$\text{mín}\, J(\xi) = \frac{1}{2}\xi^T H \xi + h^T \xi \quad \text{sujeto a:} \quad \begin{cases} A\xi = b; \\ C\xi \leq d \end{cases} \tag{2.1}$$

donde H es una matriz definida o semi-definida positiva para que el problema sea convexo y, por lo tanto, sea posible encontrar un mínimo global [13]. A continuación, se describen las transformaciones aplicadas para representar el problema de control como una formulación del problema QP, la cual debe resolverse para cada instante de muestreo. La representación del problema de control mediante una forma canónica QP presenta la ventaja de que se puede resolver por medio de distintos solvers disponibles, ya sea comerciales como académicos de código abierto.

2.1.1. Definición del Problema de Control

Considérese el siguiente sistema lineal discretizado e invariante en el tiempo:

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k \\ y_k &= Cx_k \end{aligned} \tag{2.2}$$

Donde $x_k \in \mathbb{R}^n$ representa el valor de los n estados del sistema en el instante de muestreo k, $y_k \in \mathbb{R}^p$ y $u_k \in \mathbb{R}^m$ son los correspondientes vectores de p salidas y m entradas de control, respectivamente. Además, $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$ y $C \in \mathbb{R}^{p\times n}$ representan las dinámicas del sistema. A describe la influencia del estado actual en el siguiente estado, B describe la influencia de la entrada actual en el siguiente estado y C describe cómo es que el estado afecta la salida. Inicialmente asumimos que todos los estados del sistema son medibles para el instante k.
Se requiere diseñar un controlador que sea capaz de mover la salida del sistema a un valor de referencia. Por tanto, se define el vector de error $e_k \in \mathbb{R}^p$ como la diferencia entre la referencia $r_k \in \mathbb{R}^p$ y la salida y_k:

$$e_k = r_k - y_k \tag{2.3}$$

Si se considera que $r_k = 0$ (se busca llevar la salida al origen), entonces obtenemos $e_k = -y_k$. Luego, para penalizar las desviaciones del punto de equilibrio, convenientemente se define una función de costo asociada al cuadrado del error:

$$J_k = y_k^\top y_k \tag{2.4}$$

Sustituyendo la expresión de y_k en (2.2) en (2.4), se obtiene:

$$J_k = x_k^\top \underbrace{C^\top C}_{\Omega} x_k = x_k^\top \Omega x_k \tag{2.5}$$

Donde $\Omega \in \mathbb{R}^{n\times n}$ es una matriz de pesos que refleja la importancia relativa de cada estado en la determinación de la salida del sistema. Adicionalmente, al considerar aspectos prácticos como el desgaste de los actuadores, restricciones físicas de las componentes del sistema, la seguridad de operación, el consumo energético, etc., es conveniente agregar un término cuadrático que penalice cambios drásticos en las entradas de control u_k, quedando la función de costo como:

$$J_k = x_k^\top \Omega x_k + u_k^\top \Gamma u_k \tag{2.6}$$

Donde la matriz $\Gamma \in \mathbb{R}^{m\times m}$ representa factores de penalización definidos a tiempo de diseño. Para el caso de interés de múltiples entradas, podemos considerar por simplicidad una matriz diagonal $\Gamma = Iq$, donde $I \in \mathbb{R}^{m\times m}$ es la matriz identidad y q es un escalar que pondera la entrada.

Dado que el control predictivo requiere calcular el costo obtenido para las futuras entradas dentro del horizonte de predicción, se generaliza el funcional de costo como la suma de los costos individuales dentro del horizonte de predicción h como:

$$J(x_k, u_k) = \sum_{k=0}^{h-1} \left(x_k^\top \Omega x_k + u_k^\top \Gamma u_k \right) + x_h^\top \Omega_h x_h \tag{2.7}$$

Donde el término $x_h^\top \Omega_h x_h$ representa un estado terminal con su correspondiente matriz de penalización $\Omega_h \in \mathbb{R}^{n \times n}$, el cual se introduce para aproximar el comportamiento de un lazo MPC con horizonte infinito (lo cual no se puede implementar en la práctica) mediante un lazo con horizonte finito. En general, la expresión terminal influencia la calidad del control logrado, la estabilidad del sistema, y la complejidad numérica del problema. La correcta selección de esta expresión es un problema no trivial, el cual se resuelve por medio de heurísticas [14] [15].

Para determinar la acción de control óptima a lo largo del horizonte, es necesario encontrar la secuencia de entradas $\vec{u}^* = \begin{bmatrix} u_0^{*T} & u_1^{*T} & \cdots & u_{h-1}^{*T} \end{bmatrix}^T \in \mathbb{R}^{(m \cdot h)}$ que minimiza el funcional de costo a lo largo del horizonte de predicción. Formalmente, y suponiendo que se dispone de una medición o estimación del vector de estados actual x_k, el problema QP se describe como:

$$\begin{aligned} \vec{u}^* &= \arg\min_{\vec{u}} J(x_k, u_k) \\ \text{sujeto a:} & \\ x_0 &= x, \\ x_{k+1} &= Ax_k + Bu_k, \\ u^{\text{mín}} &\leq u_k \leq u^{\text{máx}}, \\ x^{\text{mín}} &\leq x_k \leq x^{\text{máx}}, \\ x_h^{\text{mín}} &\leq x_h \leq x_h^{\text{máx}}. \end{aligned} \tag{2.8}$$

Donde x_0 corresponde al estado inicial cuyo valor al instante k puede ser medido o estimado mediante un observador, $u^{\text{mín}}$ y $u^{\text{máx}} \in \mathbb{R}^m$ que especifican límites para los valores mínimos y máximos para cada entrada, y $x^{\text{mín}}$ y $x^{\text{máx}} \in \mathbb{R}^n$ que especifican límites para los valores mínimos y máximos para cada estado, y $x_h^{\text{mín}}$ y $x_h^{\text{máx}} \in \mathbb{R}^n$ que especifican límites para los valores mínimos y máximos para cada estado terminal.
La formulación anterior plantea un problema de control en lazo abierto, ya que la secuencia de entradas óptimas calculadas al instante k no considera alteraciones o perturbaciones que puedan ocurrir al momento de ejecutar las acciones futuras. En el caso de ocurrir perturbaciones, la aplicación de las entradas precalculadas no generarán la trayectoria óptima en los estados y salidas, generando una degradación en el desempeño o incluso inestabilidad del sistema. Además, como se comentó previamente, la solución encontrada es inherentemente subóptima debido a que el funcional de costo es solo una aproximación para un horizonte infinito. Para tratar ambas limitaciones, se plantea el concepto de *receding horizon*, el cual plantea que *"Un controlador subóptimo de horizonte infinito puede ser diseñado resolviendo repetidamente problemas de control óptimo de tiempo finito en una forma de horizonte en retroceso (...)"* [16]. Bajo el enfoque de *receding horizon*, se resuelve un problema de optimización de horizonte finito en cada tiempo de muestreo. Una vez determinada la secuencia de entradas $\vec{u}^*$, solo se aplica el primer elemento u_0^* al sistema, descartando el resto de valores de la secuencia óptima. Luego, en el siguiente intervalo de muestreo se determinan los nuevos estados obtenidos a partir de la entrada aplicada y se define un nuevo problema de optimización con un nuevo

valor inicial, lo cual introduce un *feedback* indirecto, cerrando el lazo de control evitando que las diferencias entre el estado medido y el predicho se propaguen a instantes futuros.

Los pasos para aplicar MPC en cada instante de muestreo se resumen a continuación:

- En el instante de muestreo k se mide (o estima) el estado de la planta, x_k.
- Plantear el problema QP.
- Calcular el nuevo funcional de costo y resolver para determinar la secuencia óptima de entradas que lo minimizan, $\vec{u}^*$.
- Aplicar en la entrada el primer elemento de la secuencia de control óptima, u_0^*.
- Repetir para el siguiente instante de muestreo, $k+1$.

Matemáticamente, el problema QP a resolver en cada intervalo de control se puede plantear de distintas formas según la elección de las variables de decisión. Cuando se elige como única variable de decisión la entrada u_k, entonces se obtiene la forma denominada *Formulación Densa*, y si se eligen como variable de decisión tanto la entrada u_k como el estado x_k se obtiene la forma denominada *Formulación Sparse*. La selección de la forma de representar el problema QP normalmente influye significativamente en el costo computacional, requerimientos de memoria y tiempos de convergencia asociados al problema QP. Para este trabajo se hará uso de la formulación densa.

2.1.2. Formulación Densa

Considerando como variable de decisión a la entrada del sistema, se definen los siguientes vectores:

$$\vec{u} = \begin{bmatrix} u_0 \\ u_1 \\ \vdots \\ u_{h-1} \end{bmatrix} \quad \vec{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_h \end{bmatrix} \tag{2.9}$$

donde $\vec{u} \in \mathbb{R}^{(m \cdot h)}$ y $\vec{x} \in \mathbb{R}^{(n \cdot h)}$. Con estas definiciones podemos reescribir el sistema (2.2) en términos de $\vec{u}$ y $\vec{x}$ como:

$$\vec{x} = \underbrace{\begin{bmatrix} A \\ A^2 \\ \vdots \\ A^h \end{bmatrix}}_{\mathcal{A}} x_0 + \underbrace{\begin{bmatrix} B & 0 & 0 & \cdots & 0 \\ AB & B & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ A^{h-1}B & A^{h-2}B & \cdots & AB & B \end{bmatrix}}_{\mathcal{O}} \vec{u} = \mathcal{A}x_0 + \mathcal{O}\vec{u} \tag{2.10}$$

donde $\mathcal{A} \in \mathbb{R}^{(n \cdot h) \times n}$ y $\mathcal{O} \in \mathbb{R}^{(n \cdot h) \times (m \cdot h)}$. También expandir el funcional de costo (2.7) para dejarlo en términos de x_0 y $\tilde{\mathbf{u}}$:

$$J(x_0,\vec{u}) = x_0^T\Omega x_0 + \vec{x}^T \underbrace{\begin{bmatrix} \Omega & 0 & \cdots & 0 \\ 0 & \Omega & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \Omega_h \end{bmatrix}}_{\mathbf{Q}} \vec{x} + \vec{u}^T \underbrace{\begin{bmatrix} \Gamma & 0 & \cdots & 0 \\ 0 & \Gamma & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \Gamma \end{bmatrix}}_{\mathbf{R}} \vec{u} = x_0^T\Omega x_0 + \vec{x}^T Q\vec{x} + \vec{u}^T R\vec{u} \tag{2.11}$$

donde $\mathbf{Q} \in \mathbb{R}^{(n\cdot h)\times(n\cdot h)}$ y $\mathbf{R} \in \mathbb{R}^{(m\cdot h)\times(m\cdot h)}$. Luego, concentrando (2.10) y (2.11) se obtiene:

$$\begin{aligned} J(x_0,\vec{u}) &= x_0^T\Omega x_0 + (\mathcal{A}x_0 + \mathcal{O}\vec{u})^T Q(\mathcal{A}x_0 + \mathcal{O}\vec{u}) + \vec{u}^T R\vec{u} \\ &= x_0^T\Omega x_0 + x_0^T\mathcal{A}^T Q\mathcal{A}x_0 + x_0^T\mathcal{A}^T Q\mathcal{O}\vec{u} + \vec{u}^T\mathcal{O}^T Q\mathcal{A}x_0 + \vec{u}^T\mathcal{O}^T Q\mathcal{O}\vec{u} + \vec{u}^T R\vec{u} \end{aligned} \tag{2.12}$$

donde los términos $x_0^T\mathcal{A}^T Q\mathcal{O}\vec{u}$ y $\vec{u}^T\mathcal{O}^T Q\mathcal{A}x_0$ son iguales ya que uno es el transpuesto del otro y cada uno es de dimensiones 1×1, por lo que se puede transponer el segundo término y sumarlo al primero, resultando:

$$J(x_0,\vec{u}) = x_0^T(\Omega + \mathcal{A}^T Q\mathcal{A})x_0 + \vec{u}^T(R + \mathcal{O}^T Q\mathcal{O})\vec{u} + 2x_0^T\mathcal{A}^T Q\mathcal{O}\vec{u} \tag{2.13}$$

Se define $H \in \mathbb{R}^{(m\cdot h)\times(m\cdot h)}$ y $q^T \in \mathbb{R}^{1\times(m\cdot h)}$ como:

$$\frac{1}{2}H = R + \mathcal{O}^T Q\mathcal{O} \tag{2.14}$$

$$q^T = 2x_0^T\mathcal{A}^T Q\mathcal{O} \tag{2.15}$$

Reemplazando (2.14) y (2.15) en (2.13) se obtiene el funcional de costo:

$$J(x_0,\vec{u}) = \frac{1}{2}\vec{u}^T H\vec{u} + q^T\vec{u} \tag{2.16}$$

Nótese que el término $x_0(\Omega_h + \mathcal{A}^T Q\mathcal{A})x_0$ no aparece en el funcional de costo (2.16) ya que para cada instante de muestreo su valor es constante y por lo tanto no va a afectar la búsqueda de un $\vec{u}$ que minimice el costo.
Finalmente se reescriben el resto de las restricciones en términos de $\vec{u}$ y $\vec{x}$ como se describe a continuación:

a) Limitaciones en la señal de estado:

$$\begin{aligned} \vec{x}^{\text{mín}} &\leq \vec{x} \leq \vec{x}^{\text{máx}} \\ \vec{x}^{\text{mín}} &\leq \mathcal{A}x_0 + \mathcal{O}\vec{u} \leq \vec{x}^{\text{máx}} \\ (\vec{x}^{\text{mín}} - \mathcal{A}x_0) &\leq \mathcal{O}\vec{u} \leq (\vec{x}^{\text{máx}} - \mathcal{A}x_0) \end{aligned} \tag{2.17}$$

$$\begin{aligned} \mathcal{O}\vec{u} &\leq (\vec{x}^{\text{máx}} - \mathcal{A}x_0) \\ -\mathcal{O}\vec{u} &\leq -(\vec{x}^{\text{mín}} - \mathcal{A}x_0) \end{aligned} \tag{2.18}$$

donde:

$$\vec{x}^{\text{mín}} = \begin{bmatrix} x^{\text{mín}} \\ \vdots \\ x^{\text{mín}} \end{bmatrix} \quad \vec{x}^{\text{máx}} = \begin{bmatrix} x^{\text{máx}} \\ \vdots \\ x^{\text{máx}} \end{bmatrix} \tag{2.19}$$

tal que $\vec{x}^{\text{mín}}$ y $\vec{x}^{\text{máx}} \in \mathbb{R}^{(n \cdot h)}$.

b) Región Final:

$$\begin{aligned} x_h^{\text{mín}} &\leq x_h \leq x_h^{\text{máx}} \\ x_h^{\text{mín}} &\leq (\mathcal{A}_h x_0 + \mathcal{O}_h \vec{u}) \leq x_h^{\text{máx}} \\ (x_h^{\text{mín}} - \mathcal{A}_h x_0) &\leq \mathcal{O}_h \vec{u} \leq (x_h^{\text{máx}} - \mathcal{A}_h x_0) \end{aligned} \tag{2.20}$$

$$\begin{aligned} \mathcal{O}_h \vec{u} &\leq (x_h^{\text{máx}} - \mathcal{A}_h x_0) \\ -\mathcal{O}_h \vec{u} &\leq -(x_h^{\text{mín}} - \mathcal{A}_h x_0) \end{aligned} \tag{2.21}$$

donde:

$$\mathcal{O}_h = \begin{bmatrix} A^{h-1}B & A^{h-2}B & \cdots & AB & B \end{bmatrix}, \quad \mathcal{A}_h = A^h \tag{2.22}$$

tal que $\mathcal{O}_h \in \mathbb{R}^{n \times (m \cdot h)}$ y $\mathcal{A}_h \in \mathbb{R}^{n \times n}$.

c) Limitaciones en la señal de entrada:

$$\begin{aligned} u^{\text{mín}} &\leq u_k \leq u^{\text{máx}} \\ \vec{u}^{\text{mín}} &\leq \vec{u} \leq \vec{u}^{\text{máx}} \end{aligned} \tag{2.23}$$

donde:

$$\vec{u}^{\text{mín}} = \begin{bmatrix} u^{\text{mín}} \\ \vdots \\ u^{\text{mín}} \end{bmatrix} \quad \vec{u}^{\text{máx}} = \begin{bmatrix} u^{\text{máx}} \\ \vdots \\ u^{\text{máx}} \end{bmatrix} \tag{2.24}$$

tal que $\vec{u}^{\text{mín}}$ y $\vec{u}^{\text{máx}} \in \mathbb{R}^{m \cdot h}$.

y en forma matricial, todas las restricciones se pueden escribir como:

$$\underbrace{\begin{bmatrix} \mathcal{O}_h \\ -\mathcal{O}_h \\ \mathcal{O} \\ -\mathcal{O} \end{bmatrix}}_{G} \vec{u} \leq \underbrace{\begin{bmatrix} \vec{x}_h^{\text{máx}} - \mathcal{A}_h x_0 \\ -\vec{x}_h^{\text{mín}} + \mathcal{A}_h x_0 \\ \vec{x}^{\text{máx}} - \mathcal{A} x_0 \\ -\vec{x}^{\text{mín}} + \mathcal{A} x_0 \end{bmatrix}}_{c}, \quad \underbrace{\begin{bmatrix} u^{\text{mín}} \\ \vdots \\ u^{\text{mín}} \end{bmatrix}}_{a} \leq \vec{u} \leq \underbrace{\begin{bmatrix} u^{\text{máx}} \\ \vdots \\ u^{\text{máx}} \end{bmatrix}}_{b}. \tag{2.25}$$

donde $G \in \mathbb{R}^{(2 \cdot n + 2 \cdot n \cdot h) \times (m \cdot h)}$, $c \in \mathbb{R}^{(2 \cdot n + 2 \cdot n \cdot h)}$, $a \in \mathbb{R}^{(m \cdot h)}$ y $b \in \mathbb{R}^{(m \cdot h)}$.
Nótese que las restricciones en (2.8) se transforman en restricciones lineales en $\vec{u}$, con lo cual se define el problema de optimización QP como:

$$\vec{u}^* = \min_{\vec{u}} J(x_0, \vec{u}) \quad \text{sujeto a} \quad \begin{cases} G\vec{u} \leq c, \\ a \leq \vec{u} \leq b. \end{cases} \tag{2.26}$$

2.1.3. Seguimiento de Referencia

Para el seguimiento de una referencia $r \neq 0$ en la salida se requiere que $y_\infty \to r$, donde y_∞ representa el valor de la salida en estado estacionario. Para determinar el valor de los estados y entradas que permiten llevar la salida al valor de referencia, se reescribe la expresión en estado estacionario para el sistema en (2.2) como:

$$\begin{aligned} x_\infty &= A x_\infty + B u_\infty \\ y_\infty &= C x_\infty \end{aligned} \tag{2.27}$$

y se despeja x_∞ y u_∞:

$$\underbrace{\begin{bmatrix} I - A & -B \\ C & 0 \end{bmatrix}}_{L} \begin{bmatrix} x_\infty \\ u_\infty \end{bmatrix} = \begin{bmatrix} 0 \\ r \end{bmatrix} \tag{2.28}$$

$$\begin{bmatrix} x_\infty \\ u_\infty \end{bmatrix} = L^{-1} \begin{bmatrix} 0 \\ r \end{bmatrix} \tag{2.29}$$

donde $L \in \mathbb{R}^{(n+p) \times (n+m)}$.

Y se realiza el siguiente cambio de variables:

$$\begin{aligned} \tilde{x} &= x - x_\infty \\ \tilde{u} &= u - u_\infty \end{aligned} \tag{2.30}$$

donde $\tilde{x}$ y $\tilde{u}$ representan la desviación de los estados y entradas de los valores que permiten llevar la salida a su valor de referencia en estado estacionario.

Considerando que las variables de desviación satisfacen el mismo modelo dinámico que las variables originales, el funcional de costo en función de las variables de desviación queda dado por:

$$J_h(\tilde{x}_0, \vec{\tilde{\mathbf{u}}}) = \frac{1}{2}\vec{\tilde{\mathbf{u}}}^T H \tilde{\mathbf{u}} + q^T \vec{\tilde{\mathbf{u}}} \tag{2.31}$$

donde:

$$\begin{aligned} \frac{1}{2}H &= R + \mathcal{O}^T Q \mathcal{O} \\ \tilde{h}^T &= 2\tilde{x}_0^T \mathcal{O}^T Q \mathcal{O} \end{aligned} \tag{2.32}$$

Para encontrar las restricciones de esta nueva función de costo hay que aplicar el cambio de variables (2.30) a las restricciones originales (2.25). Se llega a:

a) Limitaciones en las entradas;

$$\begin{aligned} \vec{u}^{\text{mín}} \leq \vec{u} \leq \vec{u}^{\text{máx}} \\ \vec{u}^{\text{mín}} \leq \vec{\tilde{\mathbf{u}}} + \vec{u}_\infty \leq \vec{u}^{\text{máx}} \\ \vec{u}^{\text{mín}} - \vec{u}_\infty \leq \vec{\tilde{\mathbf{u}}} \leq \vec{u}^{\text{máx}} - \vec{u}_\infty \end{aligned} \tag{2.33}$$

b) Limitaciones en los estados:

$$\begin{aligned} \vec{x}^{\text{mín}} \leq \vec{x} \leq \vec{x}^{\text{máx}} \\ \vec{x}^{\text{mín}} \leq \vec{\tilde{\mathbf{x}}} + \vec{x}_\infty \leq \vec{x}^{\text{máx}} \\ \vec{x}^{\text{mín}} - \vec{x}_\infty \leq \vec{\tilde{\mathbf{x}}} \leq \vec{x}^{\text{máx}} - \vec{x}_\infty \\ \vec{x}^{\text{mín}} - \vec{x}_\infty \leq \mathcal{A}\tilde{x}_0 + \mathcal{O}\vec{u} \leq \vec{x}^{\text{máx}} - \vec{x}_\infty \\ \vec{x}^{\text{mín}} - \vec{x}_\infty - \mathcal{A}\tilde{x}_0 \leq \mathcal{O}\vec{u} \leq \vec{x}^{\text{máx}} - \vec{x}_\infty - \mathcal{A}\tilde{x}_0 \end{aligned} \tag{2.34}$$

$$\begin{aligned} \mathcal{O}\vec{u} \leq \vec{x}^{\text{máx}} - \vec{x}_\infty - \mathcal{A}\tilde{x}_0 \\ -\mathcal{O}\vec{u} \leq -(\vec{x}^{\text{mín}} - \vec{x}_\infty - \mathcal{A}\tilde{x}_0) \end{aligned} \tag{2.35}$$

La extensión a la región final es análoga, y finalmente se pueden representar las restricciones en estado estacionario:

$$\underbrace{\begin{bmatrix} \mathcal{O}_h \\ -\mathcal{O}_h \\ \mathcal{O} \\ -\mathcal{O} \end{bmatrix}}_{\tilde{G}} \vec{\tilde{\mathbf{u}}} \leq \underbrace{\begin{bmatrix} \vec{x}_h^{\text{máx}} - x_\infty - \mathcal{A}_h\tilde{x}_0 \\ -\vec{x}_h^{\text{mín}} + x_\infty + \mathcal{A}_h\tilde{x}_0 \\ \vec{x}^{\text{máx}} - x_\infty - \mathcal{A}\tilde{x}_0 \\ -\vec{x}^{\text{mín}} + x_\infty + \mathcal{A}\tilde{x}_0 \end{bmatrix}}_{\tilde{c}}, \quad \underbrace{\begin{bmatrix} u^{\text{mín}} - u_\infty \\ \vdots \\ u^{\text{mín}} - u_\infty \end{bmatrix}}_{\tilde{a}} \leq \tilde{\mathbf{u}} \leq \underbrace{\begin{bmatrix} u^{\text{máx}} - u_\infty \\ \vdots \\ u^{\text{máx}} - u_\infty \end{bmatrix}}_{\tilde{b}} \tag{2.36}$$

donde $\tilde{G} \in \mathbb{R}^{(2\cdot n+2\cdot n\cdot h)\times(m\cdot h)}$, $\tilde{c} \in \mathbb{R}^{(2\cdot n+2\cdot n\cdot N)}$, $\tilde{a} \in \mathbb{R}^{(m\cdot h)}$ y $\tilde{b} \in \mathbb{R}^{(m\cdot h)}$.

2.1.4. Restricción de Desigualdad

La restricción de caja en (2.36) se pasa a una de desigualdad para que todas queden en una ecuación. Primero se separa en dos ecuaciones de desigualdad:

$$\begin{aligned} \tilde{a} \leq \vec{\tilde{u}} \leq \tilde{b} \\ \tilde{a} \leq \vec{\tilde{u}}, \quad \vec{\tilde{u}} \leq \tilde{b} \\ -\vec{\tilde{u}} \leq -\tilde{a}, \quad \vec{\tilde{u}} \leq \tilde{b} \end{aligned} \tag{2.37}$$

Luego se juntan con las restricciones de desigualdad de (2.36) de la siguiente manera:

$$\begin{aligned} \begin{bmatrix} I \\ -I \end{bmatrix} \vec{\tilde{\mathbf{u}}} \leq \begin{bmatrix} \tilde{\mathbf{b}} \\ -\tilde{\mathbf{a}} \end{bmatrix}, \\ \hat{G} = \begin{bmatrix} \tilde{G} \\ I \\ -I \end{bmatrix}, \quad \hat{\mathbf{c}} = \begin{bmatrix} \tilde{\mathbf{c}} \\ \tilde{\mathbf{b}} \\ -\tilde{\mathbf{a}} \end{bmatrix}. \end{aligned} \tag{2.38}$$

De esta manera, $\vec{\tilde{\mathbf{u}}}$ queda restringido de la siguiente manera:

$$\hat{G}\vec{\tilde{\mathbf{u}}} \leq \hat{\mathbf{c}} \tag{2.39}$$

donde $\hat{G} \in \mathbb{R}^{(2 \cdot n + 2 \cdot n \cdot h + 2 \cdot m \cdot h) \times (m \cdot h)}$ y $\hat{\mathbf{c}} \in \mathbb{R}^{(3 \cdot n + 2 \cdot n \cdot h + 2 \cdot m \cdot h)}$.

2.1.5. Planteamiento final de problema QP

Finalmente, el problema QP queda planteado, de forma densa, con seguimiento de referencia r distinta a 0 y con todas las restricciones en forma de desigualdad, como encontrar $\vec{\tilde{\mathbf{u}}}^*$ tal que minimice la función de costo $J_h(\tilde{x}_0, \vec{\tilde{\mathbf{u}}})$:

$$J_h(\tilde{x}_0, \vec{\tilde{\mathbf{u}}}) = \frac{1}{2} \vec{\tilde{\mathbf{u}}}^T H \vec{\tilde{\mathbf{u}}} + \tilde{\mathbf{q}}^T \vec{\tilde{\mathbf{u}}} \tag{2.40}$$

donde:

$$\begin{aligned} \frac{1}{2} H &= R + \mathcal{O}^T Q \mathcal{O}, \\ \tilde{\mathbf{q}}^T &= 2 \tilde{x}_0^T \mathcal{A}^T Q \mathcal{O}, \\ \text{sujeto a} \quad \hat{G}\vec{\tilde{\mathbf{u}}} &\leq \hat{\mathbf{c}}. \end{aligned} \tag{2.41}$$

Definiremos la dimensión del problema QP como NxM, donde N representa el número de variables de entrada y M el número de restricciones.

$$\text{Donde: } \mathrm{N} = h \cdot m \quad \text{y} \quad \mathrm{M} = 2h(a + b)$$

2.1.6. Resultado Final

Una vez encontrado el vector $\vec{\tilde{\mathbf{u}}}^*$ es necesario revertir el cambio de variables (2.30) que se hizo para seguir la referencia y así obtener las entradas que son apropiadas para aplicar en la planta. Esto se hace de la siguiente manera:

$$\vec{\mathbf{u}}^* = \vec{\tilde{\mathbf{u}}}^* + \vec{\mathbf{u}_\infty} \tag{2.42}$$

El resultado final de MPC es el vector $\vec{\mathbf{u}}^*$ que contiene las entradas para h instantes de tiempo que son necesarias para llevar la salida de la planta a la referencia deseada. Solo se aplica la primera entrada u_0^* a la planta y en el siguiente instante de muestreo se repite el proceso.

2.2. Solver `quadprog`

En MATLAB, `quadprog` es una función del *Optimization Toolbox* que resuelve problemas de programación cuadrática convexa. Basado en la formulación previamente expuesta, estos problemas se expresan generalmente como:

$$\begin{aligned} &\min_{\vec{\mathbf{u}}} \quad \left(\frac{1}{2}\vec{\mathbf{u}}^T H \vec{\mathbf{u}} + \tilde{\mathbf{q}}^T \vec{\mathbf{u}}\right) \\ &\text{sujeto a: } \hat{M}\vec{\mathbf{u}} \leq \hat{\mathbf{c}}, \quad \tilde{a} \leq \vec{\mathbf{u}} \leq \tilde{b} \end{aligned} \tag{2.43}$$

Donde:

- H: es una matriz cuadrada, simétrica y definida positiva que representa los coeficientes de los términos cuadráticos de la función objetivo.
- $\tilde{\mathbf{q}}$: es un vector columna que contiene los coeficientes lineales de la función objetivo.
- $\hat{M}$ y $\hat{\mathbf{c}}$: representan las restricciones lineales del tipo desigualdad ($Ax \leq b$).
- $\tilde{a}$ y $\tilde{b}$: corresponden a los límites inferior y superior de las variables, respectivamente.

En este solver, se elige por defecto un algoritmo de punto interior, esto quiere decir que el algoritmo penaliza matemáticamente acercarse demasiado a los límites impuestos por las restricciones. Este método pertenece a la clase de algoritmos de optimización convexa que emplean funciones de barrera para manejar restricciones, permitiendo una búsqueda eficiente a lo largo del interior del dominio factible. El uso de funciones de barrera consiste en incorporar las restricciones del problema directamente en la función objetivo, penalizando aquellas soluciones que se aproximan a las fronteras del conjunto factible [17].

Se toma este solver como base de referencia para comparar los resultados iniciales obtenidos con `OSQP`. En el repositorio [11] se incluye un ejemplo simple de su uso, además de incluir el generador de problemas aleatorios que se asegura de generar problemas que converjan tanto con el algoritmo de `quadprog` como con el de `OSQP`.

2.3. Solver OSQP

OSQP [1] es un solver de propósito general para problemas QP, que presenta implementación para lenguaje C. Su diseño se basa en el método de dirección alternada de los multiplicadores (ADMM) [18], dividiendo el problema en componentes más simples mediante la introducción de variables auxiliares y duales. El algoritmo reformula el problema QP inicial agregando penalizaciones por restricciones a la función objetivo y establece ecuaciones iterativas para recalcular las variables primal (x_k), auxiliar (z_k) y dual (y_k). La iteración continúa hasta que se alcanza un número máximo de iteraciones o hasta que las normas de los residuos primal y dual 2.44 2.45 (r_{prim} y r_{dual}) cumplen con tolerancias predeterminadas.

Algoritmo 1 Algoritmo ADMM de OSQP

1: Given initial values x^0, z^0, y^0 and parameters $\rho > 0, \sigma > 0, \alpha \in (0, 2)$
2: **repeat**
3: $(\tilde{x}^{k+1}, \nu^{k+1}) \leftarrow$ solve linear system:

$$\begin{bmatrix} P + \sigma I & A^T \\ A & -\rho^{-1} I \end{bmatrix} \begin{bmatrix} \tilde{x}^{k+1} \\ \nu^{k+1} \end{bmatrix} = \begin{bmatrix} \sigma x^k - q \\ z^k - \rho^{-1} y^k \end{bmatrix}$$

4: $z^{k+1} \leftarrow z^k + \rho^{-1}(\nu^{k+1} - y^k)$
5: $x^{k+1} \leftarrow \alpha \tilde{x}^{k+1} + (1 - \alpha) x^k$
6: $z^{k+1} \leftarrow \Pi(\alpha z^{k+1} + (1 - \alpha) z^k + \rho^{-1} y^k)$
7: $y^{k+1} \leftarrow y^k + \rho(\alpha z^{k+1} + (1 - \alpha) z^k - z^{k+1})$
8: **until** termination criterion is satisfied

Como se describe en la literatura base, estas normas definidas para medir la proximidad a un punto óptimo dependen en gran medida de los parámetros del algoritmo (ρ, σ, α), los cuales ponderan las restricciones en la función objetivo. Estos parámetros son cruciales para la convergencia y la velocidad del solver, sin embargo no existe una estrategia universalmente óptima para seleccionarlos.

$$\begin{aligned} r^k_{\text{prim}} &= Ax^k - z^k, \\ r^k_{\text{dual}} &= Px^k + q + A^T y^k. \end{aligned} \tag{2.44}$$

$$\begin{aligned} r^k_{prim} &= \hat{M} x^k - z^k \\ r^k_{dual} &= Hx^k + \hat{h} + \hat{M}^T y^k \end{aligned} \tag{2.45}$$

En su configuración predeterminada, OSQP realiza un máximo de cuatro mil iteraciones, un valor determinado por el parámetro `max_iter`. Sin embargo, el número de iteraciones puede ser menor si se cumple el criterio de convergencia establecido. Además, existen otros parámetros por defecto relacionados con el algoritmo ADMM, como ρ, σ y α, cuyos valores están especificados en la Tabla de parámetros 2.1 y pueden ser ajustados por el usuario según sea necesario.

Estos parámetros pueden ser afinados de manera más precisa para adaptarse a las características de un problema QP particular, lo que puede resultar en un mejor desempeño del

solver. En la documentación del solver [19] se describen estos parámetros, y a continuación se resumen los más relevantes:

- **Warm Start**: Habilita utilizar warm-start, es decir, iniciar el algoritmo desde la solución del problema QP previamente resuelto.
- **Parámetro** ρ: Es el parámetro de penalización primal en el esquema ADMM implementado por `OSQP`. Controla la importancia relativa de cumplir las restricciones $Ax \in [l, u]$ durante las iteraciones del solver. Un valor bajo de ρ puede ralentizar la reducción del residuo primal, mientras que un valor alto puede dificultar la reducción del residuo dual. `OSQP` utiliza por defecto un ajuste dinámico de este parámetro para equilibrar ambos residuos y mejorar la velocidad de convergencia.
- **Parámetro** α: Controla el paso de relajación en el algoritmo ADMM. Es un valor que se encuentra en el rango de 0 a 2. Comúnmente, se utiliza un valor en el intervalo $1.5 \leq \alpha \leq 1.8$. Sin embargo, valores demasiado altos pueden generar inestabilidad, mientras que valores demasiado bajos pueden ocasionar que el número de iteraciones disponibles no sea suficiente para alcanzar la convergencia.
- **Parámetro** `eps_abs` y `eps_rel`: La precisión absoluta y precisión relativa definen las tolerancias de parada del algoritmo. Se comparan contra los residuos primal, dual y el gap de dualidad. Tolerancias más bajas pueden generar soluciones más precisas, pero mayor número de iteraciones.
- **Parámetro** `check_termination`: Establece la frecuencia con la que se verifica si se ha alcanzado el criterio de parada. Es importante tener en cuenta que realizar estas verificaciones con demasiada frecuencia puede incrementar el tiempo total de ejecución, ya que cada comprobación consume recursos y puede ralentizar el proceso de obtener la solución. Por otro lado, si las verificaciones se hacen con poca frecuencia, pueden pasar iteraciones del algoritmo que sean innecesarias y que no se puedan detectar a tiempo. Esto podría retrasar el proceso, ya que no se identifica de forma óptima cuándo el algoritmo ya ha completado el número de iteraciones necesario para llegar a la solución, lo que hubiera permitido detener el proceso antes. En resumen, la frecuencia de las revisiones debe mantenerse en un balance adecuado para evitar ambos extremos.
- **Parámetro** `scaling`: Controla el número de iteraciones utilizadas en la etapa de escalamiento previo. Esto es útil en problemas donde las variables esten en rangos de magnitud alejados. Si no se escala adecuadamente, el algoritmo de optimización podría no ser capaz de manejar estas diferencias de magnitud de manera eficiente y llevar a problemas de convergencia.
- **Parámetro** σ: Mejora la estabilidad numérica del algoritmo. Se utiliza internamente en el sistema lineal que `OSQP` resuelve en cada iteración, ayudando a evitar problemas cuando el problema está mal condicionado.
- **Parámetro** `time_limit`: Define un tiempo máximo de ejecución del solver, útil en entornos en tiempo real o embebidos.
- **Verbose**: Permite activar o desactivar el despliegue de información en pantalla durante la ejecución del solver.

Tabla 2.1: Parámetros de `OSQP` por defecto

Parámetros	**Descripción**	**Valores permitidos**	**Valor default**
`warm_start`	Realizar warm-start.	True/False	True
`verbose`	Imprimir datos e información.	True/False	False
`alpha` α	ADMM overrel parameter.	$0 < \alpha < 2$	1.6
`rho` ρ	ADMM rho step.	$0 < \rho$	0.1
`sigma` σ	ADMM sigma step.	$0 < \sigma$	10^{-6}
`scaling`	Iteraciones de escalamiento.	0 (off) ó $\mathbb{Z}^+$	10
`eps_abs`	Tolerancia absoluta.	$\mathbb{R}_0^+$	10^{-3}
`eps_rel`	Tolerancia relativa.	$\mathbb{R}_0^+$	10^{-3}
`check_termination`	Verificar término.	0 (off) ó $\mathbb{Z}^+$	25
`time_limit`	Tiempo de ejecución máximo.	0 (off) ó $\mathbb{R}_0^+$	-

3 Caracterización de tiempos de ejecución modificando parámetros de OSQP

En este capítulo se presenta una comparación entre los solvers OSQP y quadprog de MATLAB para la resolución de un conjunto de problemas QP utilizando sus configuraciones por defecto, con el objetivo de verificar la equivalencia funcional de las soluciones obtenidas. Posteriormente, se modifican los parámetros de OSQP para explorar configuraciones alternativas y analizar sus efectos en el proceso de resolución.

El análisis se sustenta en la generación automática de instancias de problemas QP en MATLAB, su traducción a C y la ejecución de dichas instancias en C.

3.1. Metodología de Exploración

Los problemas de optimización utilizados en las pruebas fueron generados a partir de un script de MATLAB disponible de trabajos previos, el cual fue modificado para obtener 100 instancias distintas para cada dimensión de problema a evaluar. Este script se encuentra disponible en el repositorio [11].

El flujo de trabajo para realizar la exploración se resume en la Figura 3.1 y las etapas se describen a continuación.

- **Generación y resolución de problemas QP:** En el script de MATLAB se define el rango de dimensiones de problemas a evaluar y la cantidad de problemas a generar por cada dimensión dentro del rango. La dimensión se especifica como NxM, como se describe en la Sección 2.1.5. Es importante destacar que, para los mismos datos de entrada, OSQP y quadprog pueden generar soluciones distintas debido a que utilizan algoritmos diferentes. Esto puede hacer que uno de los solvers tome un mínimo local como solución o hacer que diverja. De todos los problemas generados, se selecciona un subconjunto que produzca soluciones equivalentes con ambos solvers, los cuales serán adaptados a C para evaluaciones adicionales. Para los efectos de este trabajo, esta exploración se realiza de manera iterativa hasta obtener una muestra de 100 problemas que converjan con ambos solvers para cada dimensión evaluada. Los datos de los problemas seleccionados se almacenan en matrices dentro de archivos `.mat` para su procesamiento posterior.
- **Adaptación de problemas seleccionados para ejecución en entorno C:** Los

archivos .mat son procesados mediante un script de Python que genera un archivo C independiente por cada problema QP, donde se convierte la información de las matrices a formato 'Compressed Sparse Column' (CSC), además de agregar los headers necesarios para la correcta compilación en C.

- **Resolución de problemas compilados en C.** Los archivos .c generados en la etapa anterior son compilados automáticamente mediante scripts realizados en PowerShell en el caso de Windows y Bash en el caso de Linux. La elección de distintos lenguajes de scripting se debe a que se usó lo que está disponible de forma nativa en los dispositivos ocupados, pero no tiene relevancia para los tiempos de ejecución obtenidos, ya que lo que hacen estos scripts es indicarle al sistema operativo que abra, compile y ejecute los diversos problemas, pero este tiempo de compilación no es parte del tiempo total de resolución. Una vez ejecutados los problemas, se almacena la información de tiempo de ejecución y valor final de la solucion al problema primal en archivos .txt, usando formato de separación por comas para su posterior análisis.

Los scripts usados en el flujo de trabajo, así como una explicación más detallada de que contiene cada uno se puede encontrar en el repositorio [11] en la sección 'flujo de trabajo'.

3.1.1. Métricas de interés

Aunque el foco de este trabajo es la caracterización de los tiempos de ejecución, resulta necesario verificar la equivalencia funcional entre `OSQP` y `quadprog` en términos del valor de la solución primal obtenida por ambos. Si dichos valores no son suficientemente cercanos, cualquier mejora en tiempo carece de sentido. Con este propósito, se definen las métricas de interés que se emplearán en el análisis posterior.

El **Error Promedio Porcentual (EPP)** se define como el promedio del error absoluto entre los valores de la solución primal obtenidos por la configuración bajo evaluación (V_j) y los de referencia ($V_{\text{ref j}}$) expresado en porcentaje, según 3.1. El subíndice j denota cada instancia o problema individual dentro del bloque analizado, de tamaño p. Dado que se generan 100 instancias por cada dimensión de problema, el valor de p corresponde a 100.

$$\text{EPP} = \frac{\sum_{j=1}^{p} |V_j - V_{\text{ref j}}|}{p \cdot \left| \sum_{j=1}^{p} V_{\text{ref j}} \right|} \cdot 100 \tag{3.1}$$

De forma complementaria, el **Error Máximo Porcentual (EMP)** corresponde al mayor error absoluto encontrado en el bloque de problemas, como se indica en 3.2:

$$\text{EMP} = \text{máx} \left(\frac{|V_j - V_{\text{ref j}}|}{|V_{\text{ref j}}|} \cdot 100 \right) \quad \text{para} \quad j = 1, 2, \ldots, p \tag{3.2}$$

Finalmente, para comparar los tiempos de ejecución entre solvers o configuraciones, se define el **Speedup** S, calculado como la razón entre el tiempo de la referencia (T_{ref}) y el del caso evaluado (T), según 3.3. Esta métrica puede promediarse sobre múltiples ejecuciones para obtener un valor representativo.

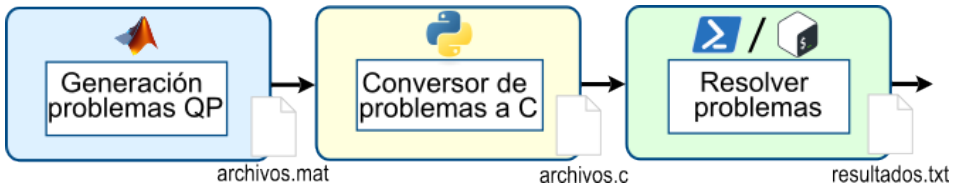

Figura 3.1: Flujo de trabajo

Tabla 3.1: Comparación de precisión de relojes en Linux y Windows.

Sistema	Frecuencia aproximada del temporizador	Resolución por tick
Linux (Raspberry Pi 4)	5.4 MHz	$\approx$ 185 ns
Linux (Raspberry Pi 5)	13.5 MHz	$\approx$ 74 ns
Windows 11 (PC)	10 MHz	$\approx$ 100 ns

$$S = \frac{T_{\text{ref}}}{T} \tag{3.3}$$

Para `quadprog`, el tiempo de resolución se midió utilizando las funciones `tic-toc` de MATLAB. Al ejecutarse dentro del entorno de MATLAB, esta medición puede verse afectada por la carga del sistema operativo y la actividad de otros procesos, los cuales pueden introducir pequeñas latencias. No obstante, este método resultó ser el más adecuado para los fines de este estudio. Además, dado que se resolvieron múltiples instancias de problemas QP, se espera que el efecto de dichas latencias se atenúe al promediar los tiempos obtenidos.

En el caso de `OSQP`, el tiempo de ejecución se registró mediante su propio sistema interno de medición (`run_time`) integrado en la librería. Tanto `OSQP` como `quadprog` emplean relojes proporcionados por el hardware del sistema. En los equipos utilizados, un computador portátil con Windows 11 y dos Raspberry Pi con Linux, dichos relojes ofrecen una resolución en el orden de los nanosegundos como se muestra en la Tabla 3.1. Dado que los tiempos mínimos reportados se encuentran en el rango de los microsegundos, esta resolución se considera suficientemente precisa para los propósitos de este estudio.

Para verificar esta precisión, se identificó qué reloj utilizan internamente las funciones `tic-toc` y `run_time`, y se ejecutó un script destinado a consultar la presición del reloj del sistema. Este script se encuentra disponible en el repositorio [11], en la sección denominada *"reloj del sistema"*. Es importante considerar que los valores obtenidos usando estas técnicas son altamente dependientes de las capacidades computacionales de la plataforma donde se ejecute el script.

3.2. Evaluaciones Preliminares de Tiempo de Ejecución y Precisión de la Solución

Este estudio preliminar tiene como propósito ofrecer un primer acercamiento al uso del solver `OSQP`, apuntando a familiarizarse con su funcionamiento y obtener una referencia inicial para facilitar futuras pruebas experimentales. De este modo, se apunta a establecer un *baseline* para el desempeño esperado, que sirva de referencia para evaluar ganancias mediante optimizaciones posteriores.

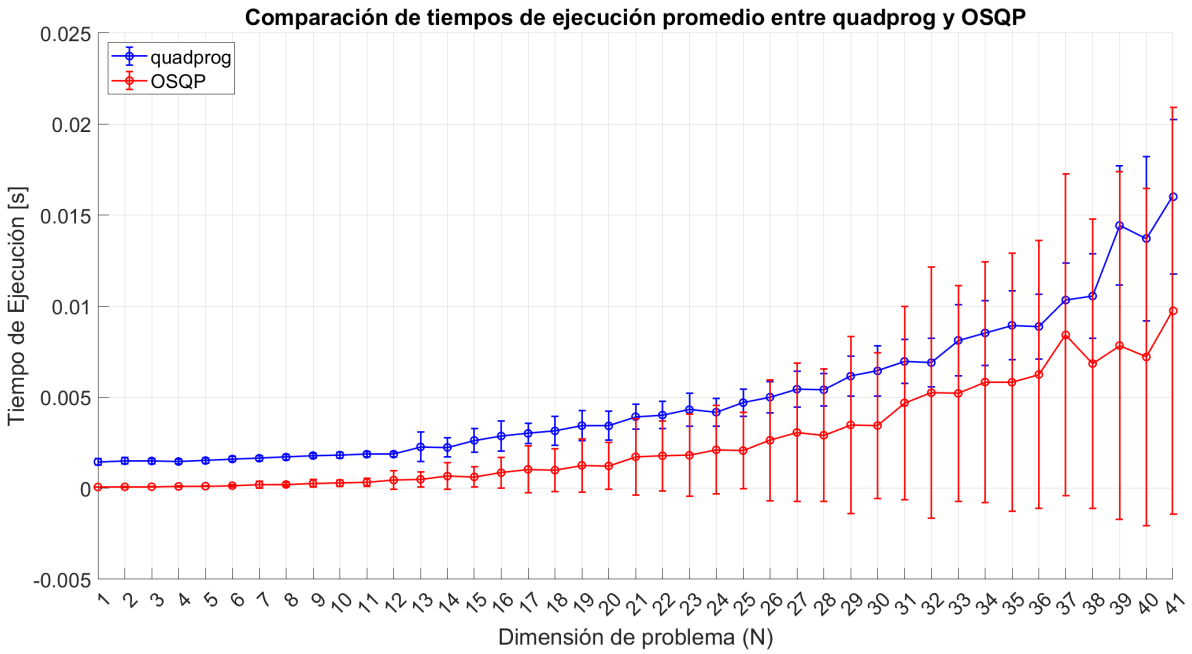

Figura 3.2: Comparación tiempo de ejecución promedio entre `quadprog` y `OSQP` con una desviación estándar.

Las pruebas se llevaron a cabo en una computadora portátil con Windows 11, equipada con 8 GB de RAM DDR5 a 4800 MHz y un procesador Intel Core i5-12450H de 2.0 GHz. Para el procesamiento, se utilizó un único núcleo, tal como se detalla en la sección de Alcances y Contribuciones 1.3. En cada experimento, se modificó de manera individual uno de los parámetros listados en la Tabla 2.1, manteniendo constantes los demás valores por defecto y sin aplicar optimizaciones. Este enfoque permite analizar el impacto específico de cada parámetro en el tiempo de resolución y en el valor de la solución primal.

La primer prueba consistió en generar 100 problemas QP por dimensión a analizar que obtengan convergencia tanto en `OSQP` como en `quadprog`. Al analizarse 41 dimensiones se obtienen 4100 problemas. De estos problemas se extrajo el EPP, EMP y S para cada dimensión con los parámetros de los solvers en configuración por defecto.

A continuación, se presentan los resultados enfocados en los tiempos de ejecución y en el error de la solución primal obtenida por `OSQP`, en comparación con el solver `quadprog` de MATLAB. Si bien es cierto las dimensiones de los problemas son del tipo NxM, para una mejor visualización de los gráficos en el eje horizontal sólo se pondrá el valor de N.

La evaluación considera tanto los tiempos de ejecución como el valor obtenido en la solución primal, priorizando un enfoque desde la perspectiva de un problema de optimización, más que de control.

Como se explica en la sección de Metodología 3.1, la comparación entre `quadprog` y `OSQP` realizada en el computador portátil puede no ser completamente precisa. No obstante, en esta etapa inicial, lo relevante es identificar la tendencia general más que centrarse en los valores específicos.

Los resultados de la Figura 3.2 muestran el valor promedio de tiempo por dimensión de

Tabla 3.2: Tiempos promedio y desviación estándar de quadprog y OSQP por dimensión, y speedup de OSQP respecto a quadprog.

	quadprog		OSQP		
Dimensión NxM	**Tiempo [µs]**	**Desv. std**	**Tiempo [µs]**	**Desv. std**	S
1x14	1448.6	168.41	64.28	3.9031	22.536
7x50	1646.0	138.35	193.11	180.77	8.5238
13x86	2269.0	803.06	488.62	404.25	4.6438
19x122	3435.8	826.42	1255.0	1461.1	2.7376
25x158	4697.9	752.59	2066.4	2108.1	2.2734
31x194	6962.4	1196.5	4689.3	5307.9	1.4847
33x206	8120.0	1949.1	5208.8	5925.1	1.5589
37x230	10337.0	2032.0	8419.1	8835.8	1.2277

Tabla 3.3: Comparación conjunta de EPP y EMP de la solución primal de OSQP respecto a quadprog.

Dimensión NxM	**EPP**	**EMP**
1 x 14	0.012	0.095
7 x 50	0.065	0.405
13 x 86	0.107	0.366
19 x 122	0.128	0.571
25 x 158	0.146	0.428
31 x 194	0.193	4.682
33 x 206	0.247	7.039
37 x 230	0.201	2.231

problema y una desviación estándar. En la figura se puede observar que, a medida que aumenta la dimensión del problema, también lo hace el tiempo de ejecución requerido para resolverlo, siguiendo una tendencia de crecimiento aproximadamente lineal. También se observa que si bien OSQP tiene un mejor rendimiento promedio, tambien presenta una desviacion estandar mayor, lo cual podria ser relevante para aplicaciones de control en tiempo real que necesiten resolver el problema QP en un tiempo acotado.

En la Tabla 3.2 se observa que OSQP presenta una mejora en los tiempos de resolución promedio para todas las dimensiones, comenzando con un speedup de 22 para problemas de dimensión 1x14 y que se reduce progresivamente a medida que crece el tamaño del problema. Los resultados completos para todas las dimensiones analizadas se encuentran en la Tabla A.1 ubicada en el Anexo.

En la comparación de la solución entre ambos solvers realizada en la Tabla 3.3, se observa que, en términos de solución primal, ambos solvers pueden considerarse equivalentes. A medida que aumenta la dimensión del problema, los EPPs tienden a mantenerse en valores alrededor del 0.2 %.

Los EMPs reportados permanecen por debajo del 1 % para las primeras dimensiones, pero aumentan conforme lo hace la dimensión del problema. El mayor EMP registrado es de un 7 % para los problemas de dimensión 33x206. Si es de interés consultar los datos completos de todas las dimensiones analizadas, se puede revisar la Tabla A.2 ubicada en el Anexo.

De un total de 4100 soluciones comparadas, se observa que 27 soluciones presentan una diferencia superior al 1 % entre ambos solvers, lo cual se considera despreciable respecto del total. Esta comparación se realiza para detallar las diferencias algorítmicas entre ambos solvers

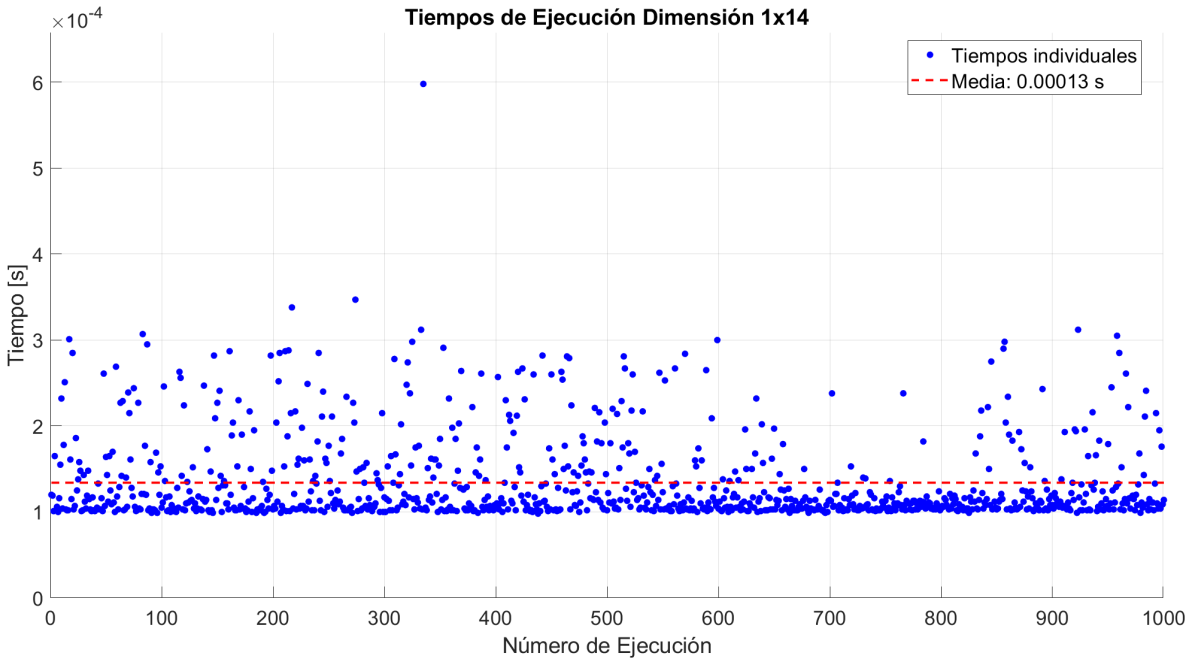

Figura 3.3: Tiempos de ejecución problema de dimensión 1x14

Tabla 3.4: Comparación de estadísticas para dimensiones 1x14 y 41x254

Estadística	Dimensión 1x14	Dimensión 41x254
Tiempo promedio [s]	$1.34 \cdot 10^{-4}$	$6.98 \cdot 10^{-3}$
Tiempo máximo [s]	$5.98 \cdot 10^{-4}$	$1.63 \cdot 10^{-2}$
Desviación estándar	$5.00 \cdot 10^{-5}$	$1.92 \cdot 10^{-3}$

y se encuentra disponible en la sección 'comparación inicial' del repositorio [11].

Este experimento nos permite observar que en terminos de solución primal, ambos solvers obtienen resultados cercanos. Por lo tanto, para efectos prácticos, consideraremos que las soluciones son equivalentes.

En las pruebas de ajuste de parámetros, el error se comparará respecto a la base obtenida por `OSQP`, y no con `quadprog`, ya que esto permite comparar el rendimiento dentro del mismo algoritmo.

La segunda prueba realizada consistió en `OSQP` uilizar dos problemas con parámetros por defecto, excepto el número de iteraciones a realizar: uno de dimensión 1x14 que es la mínima analizada, que requiere 50 iteraciones para su resolución, y otro de dimensión 41x254 que es la máxima analizada, que requiere 400 iteraciones, cabe destacar que con parámetros por defecto se revisa si se ha llegado a la solución cada 25 iteraciones. Ambos problemas se resuelven 1000 veces, con un número fijo de 400 iteraciones, que corresponde al número de iteraciones necesarias para que el problema de mayor dimensión converja. Este experimento tiene como objetivo analizar si el tiempo de resolución está asociado únicamente con el número de iteraciones requeridas o si también depende de la dimensión del problema.

En las pruebas realizadas el tiempo promedio de resolución aumenta de $1.34 \cdot 10^{-4}$[s] como

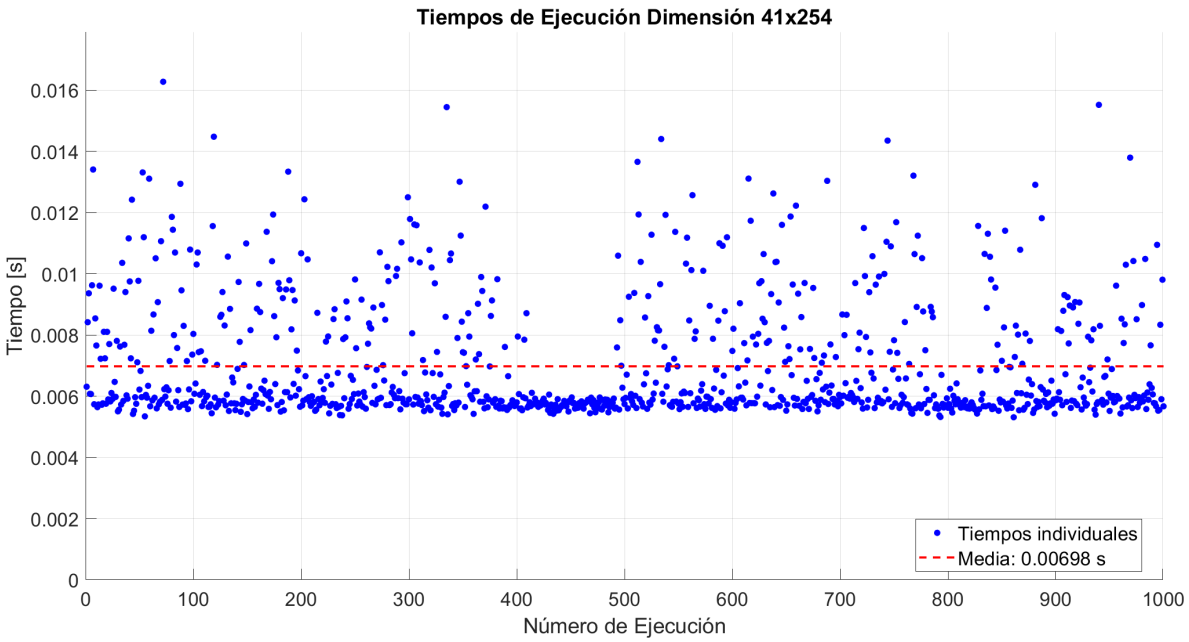

Figura 3.4: Tiempos de ejecución problema de dimensión 41x254.

se observa en la Figura 3.3 a $6.98 \cdot 10^{-}3$[s] como se observa en la Figura 3.4, así como también aumentan el tiempo máximo registrado y la desviación estándar, como se puede ver resumido en la Tabla 3.4. Las pruebas realizadas sugieren que, a medida que crece la dimensión de un problema, no sólo aumenta el número de iteraciones necesarias para resolverlo, sino también el tiempo requerido para cada iteración.

3.3. Modificación de parámetros de OSQP

En esta sección se llevarán a cabo pruebas mediante la modificación de los parámetros del solver OSQP. Se emplearán siempre los mismos problemas generados, siendo la única variación los parámetros utilizados en su resolución. Por lo tanto, la comparación de los valores de EPP, EMP y S se realizará con los resultados obtenidos utilizando los parámetros por defecto de OSQP.

3.3.1. Modificación de parámetro α

Es un factor que regula el equilibrio entre la precisión y la velocidad en la resolución de un problema QP. Si α es muy pequeño, el solver busca una solución más precisa, pero puede requerir más iteraciones en encontrarla, lo que a su vez se traduce en un aumento en el tiempo de ejecución. Si α es más grande, la solución puede ser menos precisa, pero el proceso será más rápido.

Se buscó evaluar cómo influye este parámetro en la precisión de la solución y en el tiempo de resolución, considerando además que, según la literatura de OSQP [1], valores de α mayores a 1 tienden a acelerar la convergencia, mientras que valores muy cercanos a 2 podrían generar inestabilidad. Por tanto, el objetivo es observar empíricamente si estos efectos teóricos se reflejan en los resultados prácticos al aumentar la dimensión del problema.

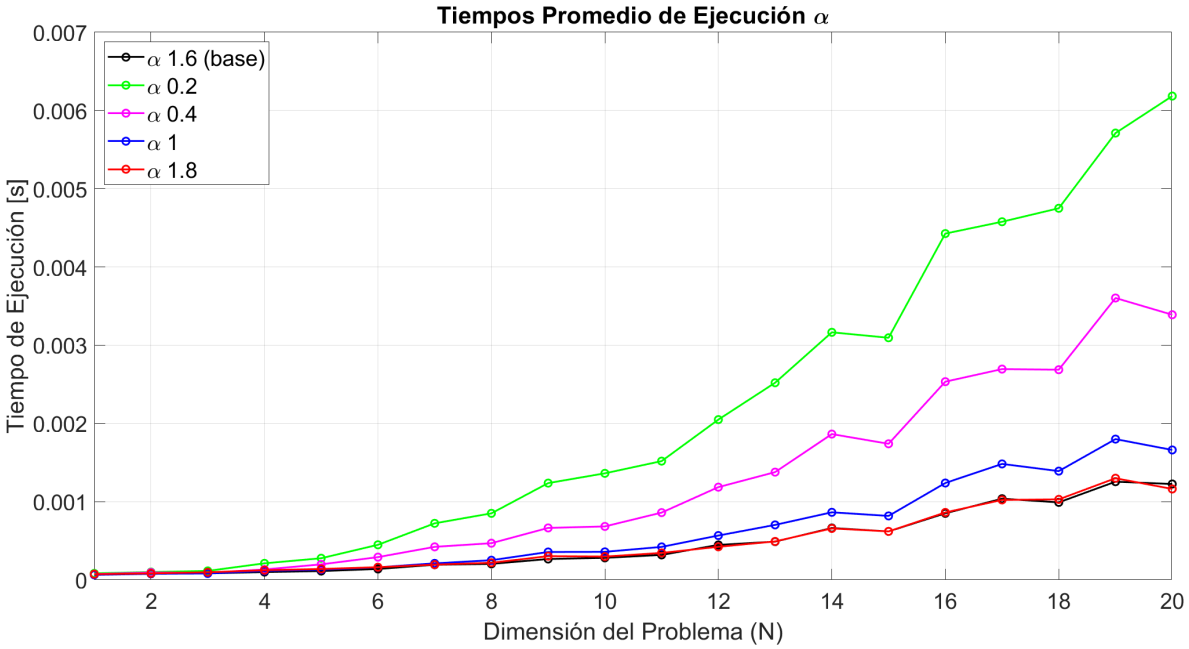

Figura 3.5: Comparación del tiempo de ejecución al modificar el parámetro α

Experimento

Para estudiar el efecto del parámetro de relajación α en el desempeño del solver OSQP, resolvimos los problemas QP generados variando únicamente el valor de α, manteniendo fijos el resto de los parámetros del solver en sus parámetros por defecto 2.1.

Los valores analizados fueron $\alpha = \{0.2,\ 0.5,\ 1.0,\ 1.6,\ 1.8\}$, cubriendo desde una configuración de sub-relajación hasta el rango superior sugerido en la literatura [1].

Análisis de Error

En las pruebas realizadas en este experimento, observamos que el parámetro α influye significativamente en la precisión de OSQP. Aunque no se identifica una tendencia clara hasta la dimensión 3x26, los valores muy bajos de α produjeron una peor precisión de manera constante a partir de esa dimensión como se puede ver en la Tabla 3.5. Para consultar los resultados completos de todos los valores, revisar la Tabla **EPP**A.4. Los errores aumentaron para valores de α bajos a medida que creció la dimensión del problema, esto sugiere que se dificulta llegar a la convergencia.

Los datos obtenidos en los experimentos muestran que, en muchos casos, se alcanza el máximo de iteraciones por defecto, por lo que se podria estar viendo afectada la obtención del valor del primal.

Por lo tanto, buscamos comprobar si α es un parámetro que impide la convergencia o si su variación sólo afecta al número de iteraciones necesarias, repetimos los experimentos con $\alpha = 0.2$ y $\alpha = 1.8$, esta vez fijando el número máximo de iteraciones en 200,000.

Al incrementar el número de iteraciones máximo, logramos que problemas que antes no

Tabla 3.5: Comparación conjunta de EPP y EMP de la solución primal según el parámetro α en OSQP.

	$\alpha = 0.2$		$\alpha = 0.4$		$\alpha = 1$		$\alpha = 1.8$	
Dimensión NxM	**EPP**	**EMP**	**EPP**	**EMP**	**EPP**	**EMP**	**EPP**	**EMP**
1 x 14	0.013	0.098	0.012	0.096	0.012	0.095	0.015	0.095
2 x 20	0.016	0.159	0.016	0.157	0.016	0.156	0.021	0.158
3 x 26	0.020	0.135	0.019	0.149	0.017	0.080	0.019	0.084
4 x 32	0.038	0.312	0.033	0.403	0.021	0.250	0.023	0.224
5 x 38	0.049	0.751	0.685	65.470	0.024	0.327	0.023	0.254
6 x 44	0.070	1.902	0.044	0.841	0.032	0.489	0.047	0.828
7 x 50	0.247	17.973	0.475	42.710	0.037	0.508	0.050	0.582
8 x 56	0.103	3.700	0.046	0.384	0.034	0.344	0.038	0.304

Tabla 3.6: Resultados de no convergencia para diferentes valores de α

configuración	Iteraciones	Resultado de no convergencia
$\alpha = 0.2$	4,000	832 veces
$\alpha = 0.2$	200,000	4 veces
$\alpha = 1.8$	4,000	87 veces
$\alpha = 1.8$	200,000	Converge siempre

convergían ahora lo hagan. En la Tabla 3.6 se tiene que en el caso de $\alpha = 0.2$, de un total de 4100, pasamos de 832 problemas sin convergencia a 4, y para $\alpha = 1.8$, de 87 problemas sin convergencia a converger todos 3.6.

A continuación, en la Tabla 3.7 evaluamos si aumentar el número máximo de iteraciones reduce los errores respecto de la configuración base de α (los datos con aumento de número de iteraciones están marcados con '(iter)' en la tabla). En general, **EPP** y **EMP** disminuyen cuando α es bajo (p. ej., $\alpha = 0.2$), mientras que la mejora es menor o nula para valores altos de α (p. ej., $\alpha = 1.8$). Esto sugiere que, con α bajos, una convergencia menos agresiva requiere más iteraciones para alcanzar el valor de la solución y, por tanto, el error se reduce al permitir más iteraciones. En cambio, con α altos, la sobre–relajación introduce pasos más agresivos que pueden aumentar las oscilaciones de los residuales y limitar la reducción del error, incluso al incrementar el número de iteraciones.

A continuación, se resumen las principales observaciones obtenidas a partir de los experimentos realizados con diferentes valores del parámetro de relajación α:

- Es importante destacar que antes de cambiar el número máximo de iteraciones, los errores máximos registrados se presentan en los valores más bajos de α, los cuales se reducen progresivamente a medida que aumenta el valor del parámetro.

- Valores alrededor de $\alpha = 1.8$, registran un mejor desempeño temporal a partir de la dimensión 22x140 en las pruebas realizadas, además de un error porcentual bajo en la mayoría de los casos.

- El uso de $\alpha = 1$ (sin relajación) no muestra ventajas significativas, salvo en problemas de muy baja dimensión. Valores mayores no solo mejoran los tiempos de ejecución, sino que también tienden a reducir el error relativo (cuando se mantienen las iteraciones por

Tabla 3.7: Comparación conjunta de EPP y EMP de la solución primal según el parámetro α en OSQP, con y sin aumento de iteraciones.

	$\alpha = 0.2$		$\alpha = 0.2$ (iter)		$\alpha = 1.8$		$\alpha = 1.8$ (iter)	
Dimensión NxM	**EPP**	**EMP**	**EPP**	**EMP**	**EPP**	**EMP**	**EPP**	**EMP**
4 x 32	0.038	0.312	0.038	0.312	0.023	0.224	0.023	0.224
6 x 44	0.070	1.902	0.051	0.788	0.047	0.828	0.047	0.828
7 x 50	0.247	17.973	0.063	0.688	0.050	0.582	0.050	0.582
16 x 104	0.643	12.747	0.042	0.341	0.032	0.313	0.032	0.313
17 x 110	0.512	11.265	0.046	0.440	0.030	0.218	0.030	0.218
35 x 218	1.622	26.099	0.043	0.931	0.023	0.367	0.033	0.938
36 x 224	1.404	25.714	0.076	2.190	0.036	0.608	0.069	2.192
39 x 242	1.458	24.317	0.142	3.966	0.055	1.657	0.136	3.969
40 x 248	1.000	12.701	0.075	1.552	0.092	2.646	0.069	2.046
41 x 254	1.578	15.197	0.093	2.945	0.046	1.148	0.087	2.948

defecto).

- En general, se observa que valores altos de α (mayores o iguales a 1.6) favorecen una mejor convergencia, mayor precisión y rapidez cuando se mantienen las iteraciones por defecto, como se observa en 3.8. Para el listado de dimensiones completo, en la Tabla A.3 ubicada en el Anexo se encuentran los speedups en más detalle.

- Basandonos en la literatura de OSQP [1] y en los experimentos realizados, se observa que valores bajos de α tienden a generar soluciones más precisas, aunque requieren un mayor número de iteraciones para converger. Por otro lado, valores altos de α pueden reducir el número de iteraciones necesarias, pero conllevan un mayor riesgo de que el algoritmo no alcance la precisión deseada y comience a oscilar, lo que implica que no se puede llegar a una solución independiente del número de iteraciones.

- Es importante advertir que acercarse demasiado al límite superior $\alpha = 2$ puede inducir inestabilidad en el algoritmo ADMM, generando oscilaciones o dificultando la convergencia, especialmente en problemas mal condicionados.

- **Recomendación:** Se sugiere mantener el valor de α en torno al valor por defecto ($\alpha = 1.6$) como punto de partida, dado que la documentación de OSQP lo establece como valor típico de uso [1].

  Si se observa que el número de iteraciones es demasiado alto para el tiempo que se tiene disponible, puede considerarse variar α en torno a 1.6 de forma gradual, monitorizando tanto la velocidad de convergencia como la presición de la solución.

  Por el contrario, si el algoritmo converge rápidamente pero la presición de la solución es insuficiente o se presentan oscilaciones, conviene reducir α para estabilizar la convergencia.

Tabla 3.8: Speedups respecto a OSQP base para distintos valores del parámetro α.

Dimensión NxM	S α 0.2	S α 0.4	S α 1	S α 1.8
1x14	0.77492	0.82505	0.99243	0.88479
2x20	0.774	0.92425	0.98722	0.93239
3x26	0.72191	0.87899	1.0172	0.85389
4x32	0.47246	0.70349	0.86994	0.80871
22x140	0.22696	0.37188	0.70307	1.0258
38x236	0.29347	0.45392	0.76177	1.0385
39x242	0.31971	0.46052	0.77748	1.0845
40x248	0.32158	0.45691	0.79364	1.0812
41x254	0.34923	0.48876	0.76998	1.0404

3.3.2. Modificación de parámetro eps_rel

`eps_rel` es la tolerancia relativa utilizada por OSQP para verificar la convergencia del algoritmo. Define el error aceptable relativo al tamaño de los residuos primal y dual. Se utiliza junto con `eps_abs` para establecer una cota de error combinada.

Valores pequeños de `eps_rel` conducen a soluciones más precisas pero requieren más iteraciones, mientras que valores más grandes permiten convergencia más rápida a costa de exactitud numérica.

Experimento

Debido al tamaño variable de los problemas se decide modificar solamente `eps_rel` y dejar `eps_abs` por defecto. Para estudiar su efecto en el desempeño del solver se resolvieron los problemas QP generados variando únicamente el valor de `eps_rel`, manteniendo fijos el resto de los parámetros del solver.

Los valores analizados fueron `eps_rel` $= \{10^{-5},\ 10^{-4},\ 10^{-3},\ 10^{-2},\ 10^{-1}\}$, comparando configuraciones que priorizan velocidad y precisión.

El objetivo del experimento fue observar cómo varía el tiempo de ejecución y la precisión de la solución al modificar esta tolerancia, considerando que un menor `eps_rel` impone condiciones de convergencia más estrictas, lo cual puede incrementar el número de iteraciones necesarias y, por ende, el tiempo total de resolución. Por el contrario, valores altos podrían reducir considerablemente el tiempo de cálculo, pero a costa de obtener soluciones con mayor error numérico.

Se analizó si estas compensaciones teóricas entre tiempo y precisión se reflejan empíricamente, y cómo se comporta el solver OSQP bajo diferentes dimensiones del problema QP.

Análisis de Error

Su ajuste tiene un impacto significativo en el tiempo de ejecución, precisión de la solución y estabilidad del solver.

- **Speedup**

  En la Tabla 3.9 se obtiene que, para dimensiones pequeñas, no necesariamente se obtiene

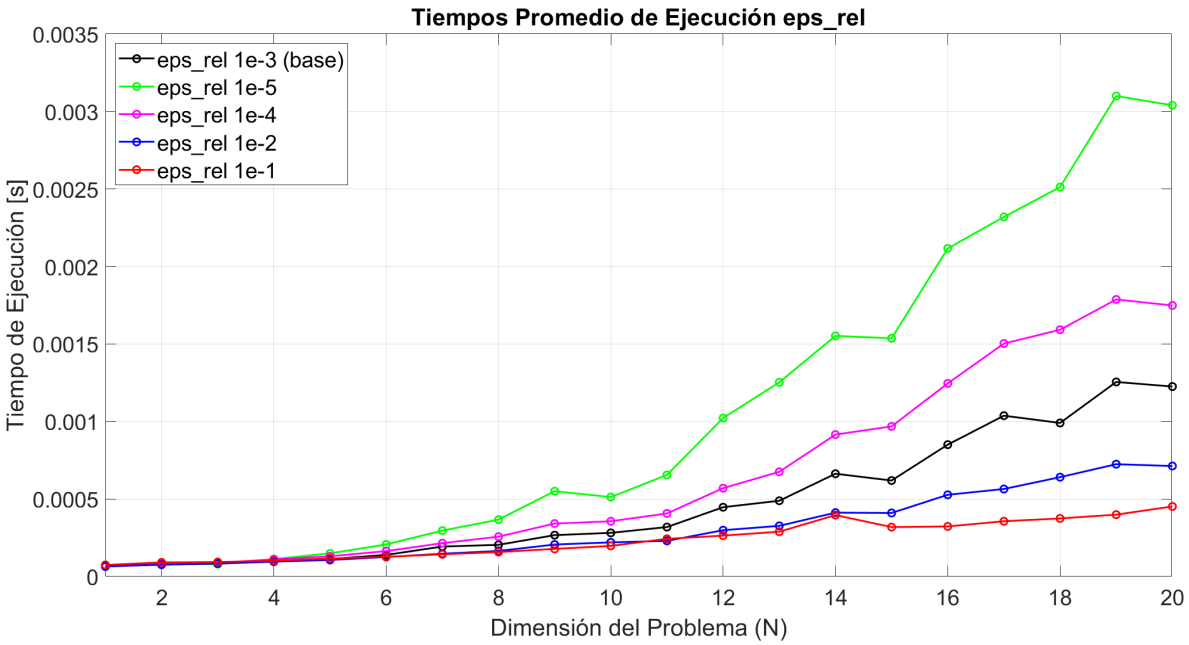

Figura 3.6: Comparación del tiempo de ejecución al modificar el parámetro eps_rel

Tabla 3.9: Speedups respecto a OSQP base para distintos valores del parámetro `eps_rel`.

Dimensión NxM	eps_rel = 1e–5	eps_rel = 1e–4	eps_rel = 1e–2	eps_rel = 1e–1
1 x 14	0.971	0.963	0.988	0.863
2 x 20	0.969	0.927	1.018	0.852
5 x 38	0.755	0.862	1.055	0.972
6 x 44	0.674	0.850	1.109	1.095
7 x 50	0.652	0.898	1.314	1.352
16 x 104	0.402	0.683	1.614	2.635
19 x 122	0.405	0.702	1.732	3.146
27 x 170	0.456	0.682	2.033	4.363
32 x 200	0.535	0.824	2.116	5.747
41 x 254	0.492	0.687	1.839	5.930

un menor tiempo al utilizar una condición de `eps_rel` menos estricta. De hecho, recién en la dimensión 6x44 se alcanza un speedup superior a 1 para `eps_rel` = 1e-1. Los tiempos más altos se presentan con el valor más pequeño de `eps_rel` = 1e-5. Una observación adicional es que, a medida que aumenta el tamaño del problema, la influencia del parámetro se vuelve más significativa: se alcanzan speedups cercanos a 6 para la elección más laxa en la dimensión 41x254, mientras que con la configuración más estricta, el speedup puede ser menor a 0,5 en esa misma dimensión, la Tabla completa se puede revisar en la sección de Anexos A.5.

- **Precisión de la solución**

  En primer lugar, se observa que los valores bajos de `eps_rel` (por ejemplo, 10^{-5} y 10^{-4}) conducen a errores bajos y estables en prácticamente todas las dimensiones del problema. En estas configuraciones, el error promedio se mantiene por cercano al 0 %, y el error máximo raramente supera el 1 %, incluso en problemas de mayor tamaño. Esto indica que una configuración estricta del criterio de tolerancia garantiza soluciones muy cercanas a las obtenidas por el solver por defecto.

Tabla 3.10: Comparación conjunta de EPP y EMP de la solución primal según el parámetro `eps_rel` en `OSQP`.

	`eps_rel` $= 10^{-5}$		`eps_rel` $= 10^{-4}$		`eps_rel` $= 10^{-2}$		`eps_rel` $= 10^{-1}$	
Dimensión NxM	**EPP**	**EMP**	**EPP**	**EMP**	**EPP**	**EMP**	**EPP**	**EMP**
1 x 14	0.012	0.095	0.011	0.095	0.009	0.097	0.009	0.097
2 x 20	0.016	0.156	0.015	0.156	0.001	0.041	0.001	0.041
3 x 26	0.017	0.080	0.017	0.080	0.002	0.067	0.002	0.067
6 x 44	0.050	0.608	0.045	0.555	0.426	5.649	4.880	106.990
10 x 68	0.077	0.336	0.071	0.316	0.731	3.315	7.835	80.261
11 x 74	0.079	0.436	0.072	0.397	0.715	3.871	5.044	51.093
14 x 92	0.369	27.285	0.088	0.476	1.112	6.642	7.485	97.016
22 x 140	0.132	0.338	0.123	0.316	1.334	4.246	9.179	49.188
24 x 152	0.190	5.245	0.123	0.327	1.298	3.247	10.516	103.340
40 x 248	0.211	2.749	0.126	0.315	1.369	5.579	10.083	54.414
41 x 254	0.231	1.693	0.128	0.332	1.503	5.222	11.415	44.347

En contraste, al aumentar `eps_rel` a valores más relajados como 10^{-2} o 10^{-1}, se aprecia una degradación progresiva en la exactitud de las soluciones. Con `eps_rel` $= 10^{-2}$, el EPP permanece en torno al 1 % en muchos casos, pero el EMP comienza a mostrar una alta dispersión, alcanzando valores superiores al 5 % en múltiples instancias. La variabilidad implica que el desempeño promedio no garantiza cercanía al óptimo en cada instancia; algunas soluciones pueden desviarse del óptimo.

Con `eps_rel` $= 10^{-1}$ los errores aumentan. El EPP supera el 10 % en problemas de alta dimensión, y el EMP puede escalar a valores que superan el 100 %. En la Tabla 3.10 se incluyen errores máximos de 106,99 % (dimensión 6x44), 103,34 % (dimensión 24x152) y 97,02 % (dimensión 14x92). Se puede revisar las tablas completa en la sección de Anexos A.6.

- **Recomendación:** Si bien valores intermedios como 10^{-2} pueden ofrecer un buen balance entre precisión y tiempo de cómputo, su uso debe ser evaluado cuidadosamente, especialmente en problemas grandes. Por otro lado, se desaconseja el uso de `eps_rel` cercano a 10^{-1} sin una validación explícita, ya que puede generar soluciones numéricamente erróneas o inestables.

  - **Alta precisión requerida**: Usar `eps_rel` $\leq 10^{-3}$, aceptando mayores tiempos de ejecución pero obteniendo errores más bajos.
  - **Aplicaciones de tiempo crítico**: Usar `eps_rel` $\geq 10^{-3}$ donde se prioriza velocidad sobre precisión, siempre evaluando la tolerancia al error.

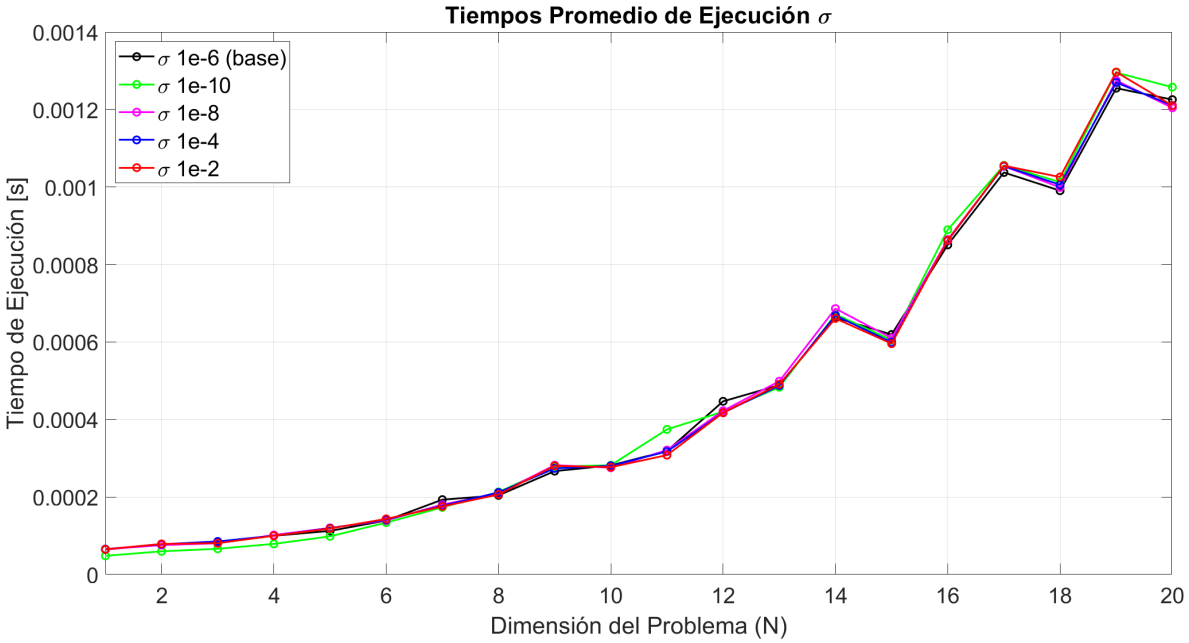

Figura 3.7: Comparación del tiempo de ejecución al modificar el parámetro σ

3.3.3. Modificación de parámetro σ

`sigma` es un parámetro de regularización dual en el solver `OSQP`. Se agrega a la matriz P como $P + \sigma I$ en el sistema lineal que se resuelve internamente en cada iteración del algoritmo ADMM [1], con el fin de mejorar la estabilidad numérica.

Se busca usar un valor pequeño de σ para que este no altere la solución final, pero si afecte la eficiencia y robustez del proceso iterativo. Valores pequeños de `sigma` preservan la fidelidad del problema original, pero pueden generar un sistema mal condicionado que ralentiza la convergencia. En cambio, valores mayores mejoran la condición del sistema y aceleran la resolución, aunque podrían introducir una leve distorsión numérica. `OSQP` emplea por defecto un valor de $\sigma = 10^{-6}$.

Experimento

Se busca observar el impacto del parámetro en el tiempo de resolución del algoritmo y en la precisión de la solución primal. Los valores evaluados fueron: $\sigma = \{10^{-10},\ 10^{-8},\ 10^{-4},\ 10^{-2}\}$ dejando los demás parámetros en sus valores por defecto [2.1].

Análisis de Error

- **Speedup**

  En el Gráfico [3.7] se observa que para $\sigma = 10^{-10}$ en dimensiones pequeñas se tiene una leve mejora en los speedups. Sin embargo, para el resto de valores no se observa un patrón claro, dado que independiente del valor de σ y de la dimensión del problema estos pueden ser mejores o peores que la solución base. Se concluye que esto es debido a que los resultados expresan valores promedio y para ver en más detalle el efecto del parámetro debería verse en un problema en particular. En la Tabla [3.11] se observan

Tabla 3.11: Speedups respecto a OSQP base para distintos valores del parámetro σ.

Dimensión NxM	$\sigma = 10^{-10}$	$\sigma = 10^{-8}$	$\sigma = 10^{-4}$	$\sigma = 10^{-2}$
1 x 14	1.336	0.991	0.981	0.990
2 x 20	1.292	1.019	0.990	0.980
11 x 74	0.851	0.993	1.004	1.035
12 x 80	1.063	1.058	1.071	1.071
13 x 86	1.011	0.980	1.002	0.995
24 x 152	0.976	0.995	1.000	0.981
25 x 158	0.999	0.989	0.993	1.004
26 x 164	0.996	1.013	1.018	1.003
39 x 242	1.004	1.021	0.995	0.991
40 x 248	1.007	1.006	1.024	1.020
41 x 254	0.999	0.988	1.000	0.994

Tabla 3.12: Comparación conjunta de EPP y EMP de la solución primal según el parámetro σ en OSQP.

	$\sigma = 10^{-10}$		$\sigma = 10^{-8}$		$\sigma = 10^{-4}$		$\sigma = 10^{-2}$	
Dimensión NxM	**EPP**	**EMP**	**EPP**	**EMP**	**EPP**	**EMP**	**EPP**	**EMP**
1x14	0.00e+00	0.00e+00	0.00e+00	0.00e+00	1.66e-06	1.66e-04	2.24e-03	9.52e-02
2x20	0.00e+00	0.00e+00	0.00e+00	0.00e+00	3.74e-06	2.55e-05	5.15e-04	8.61e-03
11x74	5.00e-08	8.00e-07	5.00e-08	8.00e-07	4.93e-06	8.28e-05	8.39e-04	3.41e-02
12x80	8.70e-07	5.09e-05	8.70e-07	5.13e-05	8.65e-05	5.07e-03	2.07e-03	9.80e-02
13x86	4.00e-08	6.80e-07	4.00e-08	6.80e-07	4.43e-06	7.15e-05	5.56e-04	1.71e-02
24x152	1.01e-06	9.59e-05	6.80e-07	6.36e-05	1.40e-04	1.35e-02	9.24e-03	8.80e-01
25x158	6.00e-08	3.34e-06	6.00e-08	3.82e-06	6.38e-06	4.02e-04	5.00e-04	1.58e-02
26x164	2.00e-07	1.18e-05	2.40e-07	1.72e-05	8.71e-06	3.64e-04	7.35e-04	3.60e-02
39x242	1.04e-06	9.12e-05	7.10e-07	5.79e-05	2.18e-05	1.29e-03	1.12e-03	8.38e-02
40x248	1.70e-07	1.51e-05	1.60e-07	1.45e-05	1.59e-05	1.29e-03	1.05e-02	1.04e+00
41x254	5.80e-07	5.55e-05	4.70e-07	4.55e-05	9.32e-06	7.70e-04	9.78e-04	8.16e-02

algunos valores de speedup. Para todos los valores ver la Tabla A.7 ubicada en el Anexo.

- **Precisión de la solución**

  Al igual que en el caso de los speedups, no se observa un patrón claro sobre la precisión de la solución, pero si se mantiene un EPPs y EMPs cercanos al 0 % 3.12, siendo el EMP máximo encontrado en la dimensión 40x248 para $\sigma = 10^{-2}$ con un valor de 1.093 %. Para todos los resultados ver la Tabla A.8 ubicada en el Anexo.

.

3.3.4. Modificación de parámetro ρ

`rho` es un parámetro que controla cuánto se prioriza el cumplimiento de las restricciones del problema durante cada iteración del algoritmo. En términos simples, si el valor de `rho` es bajo, el algoritmo es menos estricto con el cumplimiento de las restricciones y tarda más en hacer que se cumplan, haciendo que el residuo primal sea grande. Si `rho` es alto, el algoritmo intenta forzar demasiado las restricciones en cada paso, esto dificulta avanzar hacia la solución óptima y hace que aumente el residuo dual.

Por defecto, OSQP utiliza un sistema automático que ajusta dinámicamente el valor de `rho` en función del comportamiento de ambos residuos. Esto permite en teoría, encontrar un equilibrio entre precisión y velocidad de convergencia, haciendo que el proceso de optimización

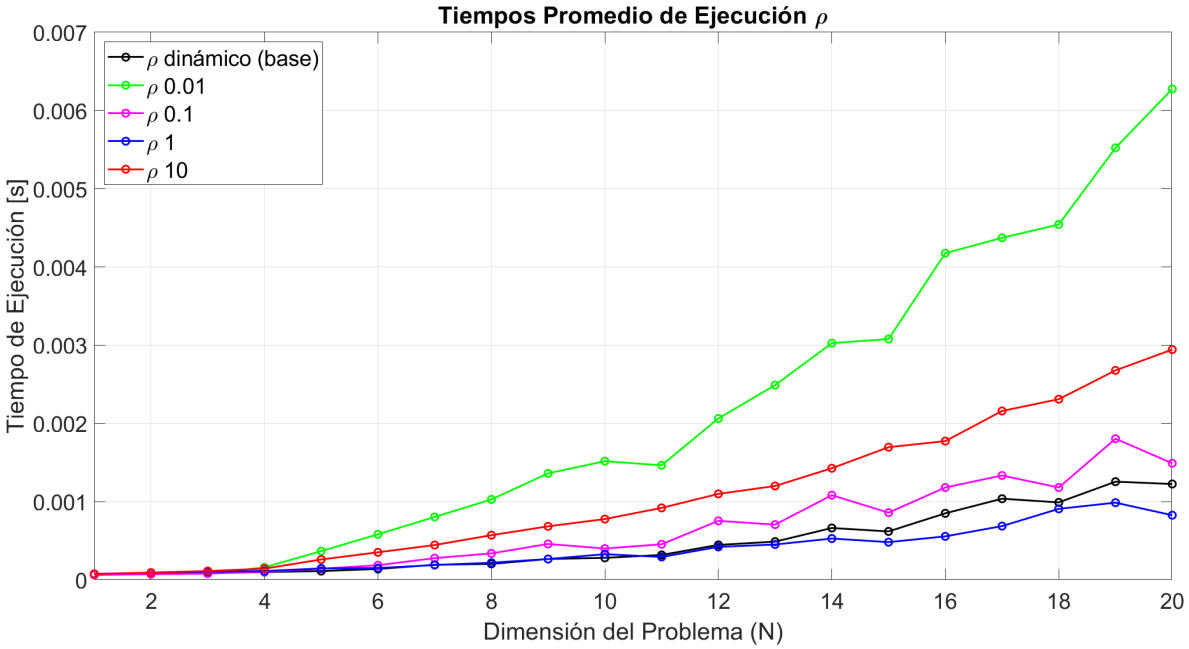

Figura 3.8: Comparación del tiempo de ejecución al modificar el parámetro ρ

sea más eficiente y estable.

Experimento

Para ver el impacto del parámetro se desactiva el ajuste dinámico del mismo y se analizan cuatro configuraciones: $\rho = \{0.01,\ 0.1,\ 1,\ 10\}$.

Análisis de Error

- **Speedup**

  La Tabla 3.13 muestra los speedups del tiempo de ejecución respecto a la configuración base del solver OSQP para distintos valores del parámetro ρ, si se desea ver todos los valores, ver la Tabla A.9 ubicada en el Anexo. A partir de estos resultados, se pueden extraer las siguientes conclusiones:

  - Para problemas de baja dimensión (por ejemplo, dimensiones 1x14, 2x20 y 3x26), se observa una ligera mejora con valores de $\rho = 0.01$ y $\rho = 0.1$, speedups mayores a 1. Sin embargo, esta tendencia se revierte en cuanto aumenta la dimensión del problema.
  - A partir de dimensiones mayores a 4x32, los speedups asociados a $\rho = 0.01$ comienzan a disminuir rapidamente, alcanzando valores por debajo de 0.3 en muchos casos.
  - El valor $\rho = 0.1$ mantiene un speedup cercanos a 0.7–0.9 a lo largo de las distintas dimensiones,
  - Por otro lado, el parámetro $\rho = 1$ muestra una tendencia creciente en los speedups conforme aumenta la dimensión del problema. En dimensiones superiores a 14x92, los speedups superan el valor de 1.5, llegando hasta valores cercanos a 2.1. Esto

Tabla 3.13: Speedups respecto a OSQP base para distintos valores del parámetro ρ.

Dimensión NxM	S ρ 0.01	S ρ 0.1	S ρ 1	S ρ 10
1x14	1.0265	1.0241	0.83157	0.86305
2x20	1.085	1.0673	0.8434	0.82891
3x26	1.0402	1.0398	0.81194	0.73128
4x32	0.62491	0.97187	0.88317	0.69594
5x38	0.30581	0.78918	0.76985	0.43161
5x44	0.23914	0.74517	0.92483	0.39547
7x50	0.24017	0.69434	1.0197	0.43356
8x56	0.19848	0.60354	0.93067	0.35742
14x92	0.21908	0.61251	1.2544	0.46454
16x104	0.20369	0.72043	1.5297	0.47963
17x110	0.23729	0.77767	1.5098	0.4806
18x116	0.21805	0.83892	1.0911	0.42887
19x122	0.22726	0.69558	1.2715	0.46868
24x152	0.24375	0.79158	1.5431	0.53255
39x242	0.32095	0.82004	1.7027	0.67778
40x248	0.31863	0.85764	1.5883	0.56608
41x254	0.34933	0.84879	1.9825	0.75754

sugiere que una elección de parámetro cercana a este valor es especialmente adecuada para problemas de gran escala.

- Finalmente, para el valor $\rho = 10$, los speedups permanecen bajo 1 en todas las dimensiones, y en muchos casos se ubican en torno a 0.4–0.7.
- En resumen, se concluye que:
  - Para problemas pequeños, valores alrededor de $\rho = 0.01$ o $\rho = 0.1$ pueden ofrecer un speedup más alto.
  - Para problemas grandes, valores cercanos a $\rho = 1$ pueden ofrecer un speedup más alto.
  - Valores cercanos a $\rho = 10$ no presentaron mejoras para ninguna dimensión evaluada.

- **Precisión de la solución**

La Tabla 3.14 presenta el EPP y EMP al variar el parámetro ρ, si es de interés ver todos los valores revisar la Tabla A.10 ubicada en el Anexo. De los experimentos reportados en la tabla se puede observar:

- Valores cercanos a $\rho = 0.01$ son los que producen mayores errores tanto promedio como máximos en casi todas las dimensiones. A partir de dimensión 5x38 en adelante, el EPP supera el 1 %, llegando a valores superiores al 4 % en dimensiones elevadas. En términos de EMP, se observan valores que superan el 90 % e incluso alcanzan más de 290 % en el caso de dimensión 14x92. Esto indica que un valor demasiado pequeño de ρ puede generar soluciones altamente inestables o imprecisas.
- Valores cercanos a $\rho = 10$ presentan EMPs mayores que valores cercanos a $\rho = 0.1$ y $\rho = 1$. Aunque el error promedio se mantiene cercano al 0 %.
- En resumen, desde el punto de vista de la precisión de la solución:
  - Valores cercanos $\rho = 0.1$ ofrecen soluciones más precisas para problemas

Tabla 3.14: Comparación conjunta de EPP y EMP de la solución primal según el parámetro ρ.

	$\rho = 0.01$		$\rho = 0.1$		$\rho = 1$		$\rho = 10$	
Dimensión NxM	**EPP**	**EMP**	**EPP**	**EMP**	**EPP**	**EMP**	**EPP**	**EMP**
1x14	0.010812	0.079137	0	0	0.011593	0.095229	0.011728	0.095229
2x20	0.0085568	0.079639	1.7e-07	1.701e-05	0.015718	0.15615	0.015717	0.15615
3x26	0.012232	0.11038	4.35e-06	0.00043476	0.016189	0.080235	0.018059	0.11431
4x32	0.077982	3.5033	0.0026046	0.25662	0.032252	0.30163	0.030126	0.25708
5x38	0.94582	48.25	0.023837	0.61404	0.034714	0.25814	0.038815	0.31652
6x44	1.2145	65.096	0.053055	2.4086	0.047099	0.60941	0.052817	0.79216
7x50	1.8472	58.164	0.11729	8.611	0.060368	0.61124	0.066271	0.5521
8x56	0.98158	30.987	0.030353	0.40551	0.060237	0.34185	0.062244	0.47639
14x92	4.0804	293.32	2.0146	198.69	0.19635	10.039	0.11874	2.4263
16x104	1.2348	19.056	0.10143	4.953	0.10803	0.47415	0.1306	0.5716
17x110	1.2	18.127	0.11482	7.2813	0.10571	0.37701	0.1203	0.45299
18x116	1.1428	42.803	0.21991	19.121	0.13786	2.0749	0.15155	2.4103
19x122	1.6397	26.997	0.14646	7.1012	0.14799	2.8155	0.13324	0.53132
24x152	1.5635	40.878	0.17175	11.546	0.19497	6.8618	0.16092	2.2765
39x242	1.7965	25.98	0.25067	11.534	0.24832	3.6403	0.27311	4.3986
40x248	1.2511	13.792	0.18825	5.1276	0.17662	1.3583	0.18789	1.7207
41x254	1.9228	16.763	0.079209	2.7525	0.22145	2.9567	0.23195	3.2133

pequeños.

- Valores cercanos a $\rho = 1$ en general ofrecen errores bajos.
- Valores cercanos a $\rho = 10$ y $\rho = 0.01$ deben evitarse si se requiere alta precisión, particularmente en problemas de gran escala.

- **Recomendación:** Si bien es posible mejorar los tiempos de ejecución sin comprometer la precisión de la solución, la opción de ajuste dinámico incluida por defecto en el solver resulta adecuada y se recomienda su uso.

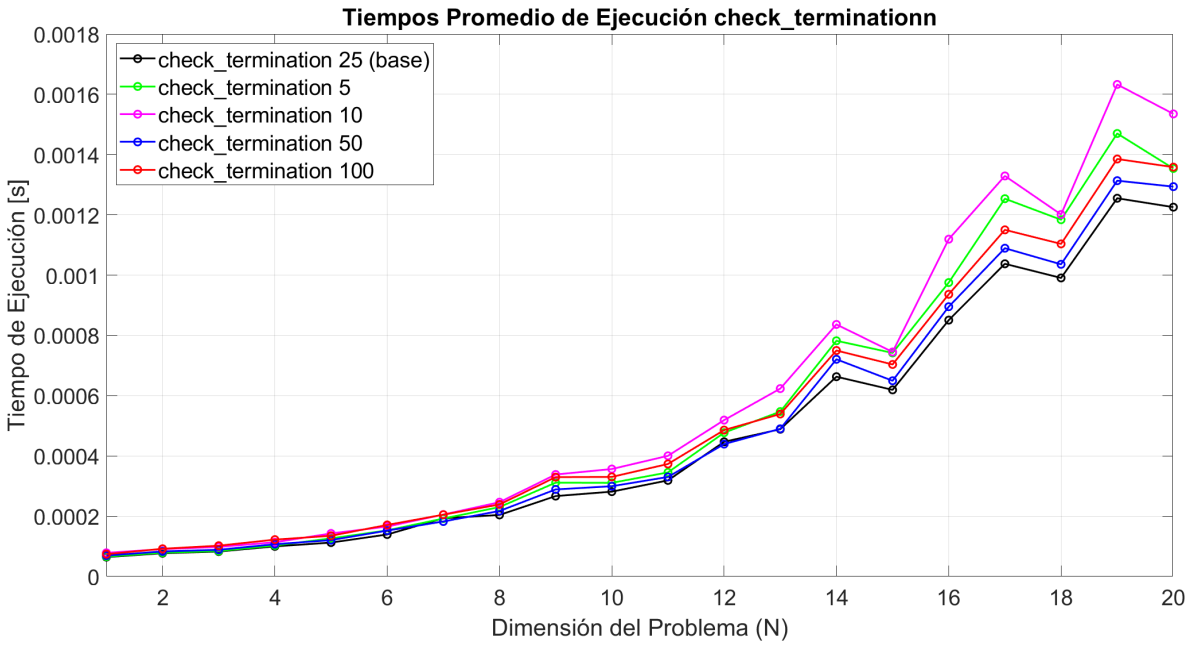

Figura 3.9: Comparación del tiempo de ejecución al modificar el parámetro check_termination

3.3.5. Modificación de parámetro check_termination

Este parámetro define cada cuántas iteraciones se verifica si el solver ha alcanzado la convergencia. Su valor por defecto es una comprobación cada 25 iteraciones. Dado que evaluar continuamente el criterio de terminación implica un costo computacional adicional, es necesario encontrar un compromiso adecuado entre la frecuencia de verificación y la realización de iteraciones innecesarias una vez alcanzada la solución.

Experimento

Se analizan cuatro configuraciones $check\_termination = \{5,\ 10,\ 50\ 100\}$. Se espera ver que a mayor sea la dimensión, sea más rentable el revisar la condición de parada a intervalos mayores.

Análisis de Error

La Tabla 3.15 presenta los speedups ejecución respecto a la configuración base del solver `OSQP`, para ver la Tabla completa revisar A.11.

A partir de los datos observados, se pueden extraer las siguientes conclusiones:

- **Speedup**

  De las pruebas realizadas, para dimensiones pequeñas (menores a 5x38), el speedup rara vez supera el valor 1 al modificar el parámetro. Se observa que el aumento del parámetro puede generar leves mejoras si la dimensión analizada también aumenta, mientras que reducir el parámetro tiende a degradar el rendimiento.

- Por otro lado, establecer `check_termination` en valores demasiado bajos (como 5 o 10) implica una revisión demasiado frecuente de los criterios de parada, lo cual introduce

Tabla 3.15: Speedups respecto a OSQP base para distintos valores del parámetro check_termination.

Dimensión NxM	S CT 5	S CT 10	S CT 50	S CT 100
1 x 14	0.976	0.802	0.928	0.867
2 x 20	0.976	0.809	0.928	0.840
3 x 26	0.981	0.778	0.935	0.811
4 x 32	0.971	0.760	0.933	0.817
5 x 38	0.890	0.714	0.933	0.833
12 x 80	0.937	0.908	1.017	0.920
13 x 86	0.893	0.864	0.997	0.906
15 x 98	0.835	0.864	0.953	0.880
16 x 104	0.872	0.859	0.950	0.908
17 x 110	0.828	0.848	0.953	0.902
27 x 170	0.794	0.963	0.934	0.904
28 x 176	0.872	0.926	1.005	0.960
29 x 182	0.826	0.853	0.952	0.893
36 x 224	0.853	0.922	1.007	0.975
40 x 248	0.875	0.920	1.008	0.965
41 x 254	0.824	0.908	1.016	0.987

Tabla 3.16: Comparación conjunta de EPP y EMP de la solución primal según el parámetro check_termination (CT) en OSQP.

	CT 5		CT 10		CT 50		CT 100	
Dimensión NxM	EPP	EMP	EPP	EMP	EPP	EMP	EPP	EMP
1x14	1.49e-03	2.29e-02	1.34e-02	9.87e-02	1.17e-02	9.52e-02	1.17e-02	9.52e-02
2x20	1.39e-04	1.04e-02	1.65e-02	1.66e-01	1.57e-02	1.56e-01	1.57e-02	1.56e-01
3x26	1.70e-04	1.35e-02	1.79e-02	8.54e-02	1.69e-02	8.00e-02	1.69e-02	8.00e-02
4x32	4.33e-02	3.59e+00	2.24e-02	2.51e-01	1.76e-02	1.16e-01	2.47e-02	2.41e-01
5x38	2.14e-02	1.99e-01	2.63e-02	1.57e-01	1.52e-02	2.40e-01	1.82e-02	2.40e-01
12x80	2.63e-02	1.26e-01	2.16e-02	1.26e-01	2.30e-02	1.36e-01	4.97e-02	2.44e-01
13x86	2.62e-02	1.56e-01	2.18e-02	1.10e-01	1.65e-02	1.27e-01	3.76e-02	1.85e-01
15x98	3.09e-02	1.94e-01	2.77e-02	1.58e-01	2.32e-02	1.46e-01	4.41e-02	2.13e-01
16x104	2.74e-02	9.54e-02	2.31e-02	9.54e-02	2.06e-02	1.41e-01	4.68e-02	2.16e-01
17x110	3.01e-02	1.79e-01	2.69e-02	1.79e-01	1.66e-02	1.35e-01	3.88e-02	2.08e-01
27x170	2.87e-02	1.26e-01	2.18e-02	9.28e-02	2.39e-02	9.65e-02	4.24e-02	1.75e-01
28x176	2.63e-02	1.31e-01	2.20e-02	1.15e-01	2.20e-02	1.31e-01	5.03e-02	1.93e-01
29x182	2.97e-02	1.46e-01	2.33e-02	1.36e-01	1.73e-02	9.85e-02	4.33e-02	1.82e-01
36x224	2.43e-02	1.32e-01	2.05e-02	1.32e-01	1.50e-02	1.06e-01	3.66e-02	1.81e-01
40x248	2.73e-02	1.12e-01	2.23e-02	1.12e-01	1.84e-02	1.33e-01	4.04e-02	2.15e-01
41x254	2.12e-02	1.10e-01	1.73e-02	1.10e-01	2.09e-02	1.43e-01	4.36e-02	3.26e-01

sobrecarga computacional sin una mejora significativa en la convergencia, resultando en speedups consistentemente por debajo de 1.

- **En resumen:**
  - Para problemas pequeños, el impacto del parámetro es menor y no se justifica modificarlo ampliamente.
  - Para problemas medianos y grandes, establecer check_termination en valores mayores al por defecto puede ofrecer tiempos más rápidos.
- **Precisión de la solución**

  Las Tabla 3.16 muestra algunos valores revelantes de EPP y EMP en relación con la solución base, para distintos valores del parámetro check_termination, si es de interés ver todos los valores revisar la Tabla A.12 ubicada en el Anexo.

A partir del análisis de los resultados, se pueden establecer las siguientes observaciones:

- En general, todos los valores del parámetro `check_termination` generan errores bajos, tanto EPP como EMP. Esto debido a que no es un parámetro que modifique internamente el algoritmo.
- **Observación particular: caso anómalo en dimensión 4x32**

  Una excepción notable se presenta en la dimensión 4x32 con el valor `check_termination = 5`, donde el EMP alcanza un valor de 3.5935 %, significativamente superior al resto de las configuraciones y dimensiones analizadas.

  Esta anomalía puede explicarse por el hecho de que verificar las condiciones de parada con excesiva frecuencia (en este caso, cada 5 iteraciones) podría inducir una detención prematura del algoritmo en situaciones donde los residuos primal y dual aún no han convergido de manera uniforme, lo cual puede hacer que una condición de parada marginalmente satisfecha detenga la optimización antes de alcanzar una solución verdaderamente cercana al óptimo.

  Además, es posible que la sensibilidad numérica del problema específico generado en esa instancia, junto con una verificación tan frecuente, haya permitido alcanzar el umbral de tolerancia sin garantizar la calidad global de la solución, lo cual se traduce en un mayor error con respecto al resultado de referencia.

  Este tipo de comportamiento no se observa de forma sistemática en otras dimensiones, lo que sugiere que no se trata de un patrón general del parámetro `check_termination`, sino más bien de un caso puntual influenciado por la interacción entre la dimensión del problema, la instancia generada y la dinámica del solver.

- **Recomendación:** Se observa que el parámetro `check_termination` no tiene un impacto crítico en la precisión final de la solución, por lo que puede ajustarse con mayor flexibilidad en función de la eficiencia computacional deseada, sin comprometer significativamente la calidad del resultado. Asimismo, según los requisitos de la aplicación, puede resultar conveniente desactivar esta comprobación y fijar un número predeterminado de iteraciones, evitando así el costo computacional asociado a la evaluación del criterio de terminación.

4 Evaluación de tiempos de ejecución en Raspberry Pi y casos de estudio de MPC

En este capítulo se busca validar la aplicabilidad práctica de las directrices obtenidas en el análisis de presentado en los capítulos anteriores, evaluando si las tendencias observadas se mantienen al ejecutar el solver `OSQP` en plataformas de cómputo embebido y en casos de estudio reales. El objetivo no es comparar tiempos absolutos entre dispositivos, pues es esperable que plataformas como la Raspberry Pi 4 presenten siempre un rendimiento inferior respecto a un computador portátil, sino determinar si las relaciones de escalamiento, sensibilidad a parámetros y comportamientos cualitativos del solver se conservan en hardware con recursos más restringidos.

En el contexto de generación y distribución eléctrica, los recursos energéticos distribuidos (DER, por sus siglas en inglés Distributed Energy Resources) son dispositivos modulares de generación y almacenamiento de energía que pueden operar de manera aislada o conectados a la red principal. Estos sistemas permiten una gestión local de la energía, contribuyendo a mejorar la eficiencia, confiabilidad y resiliencia del sistema eléctrico. En este trabajo, se incorporan dos casos de estudio relevantes para el contexto del proyecto, ambos asociados a sistemas DER y controladores MPC, lo que permite evaluar si las directrices obtenidas a partir de problemas sintéticos también se manifiestan en configuraciones reales de operación. El análisis se centra exclusivamente en la medición de los tiempos de ejecución y en la reproducción de resultados bajo distintas configuraciones, sin profundizar en el funcionamiento interno de los sistemas de control involucrados. El funcionamiento interno y el modelado detallado de los sistemas de control y simulaciones no forman parte del alcance de esta memoria, y se consideran como cajas negras, empleándose únicamente la dinámica necesaria para obtener y resolver los problemas QP asociados. Adicionalmente se realiza una breve comparación de rendimiento con un problema de dimension similar a uno implementado en FPGA.

4.1. Evaluación de tiempo de `OSQP` en Raspberry Pi

Para estas pruebas, se emplean dos sistemas embebidos: una Raspberry Pi 4 Model B Rev 1.2 con procesador ARM Cortex-A72 a 1.5 GHz y una Raspberry Pi 5 Model B Rev 1.0 con procesador ARM Cortex-A76 a 2.4 GHz.

En primer lugar, se realiza una comparación de tiempos entre la versión en C de `OSQP` ejecutada en el computador portátil y en ambas Raspberry Pi, utilizando los parámetros

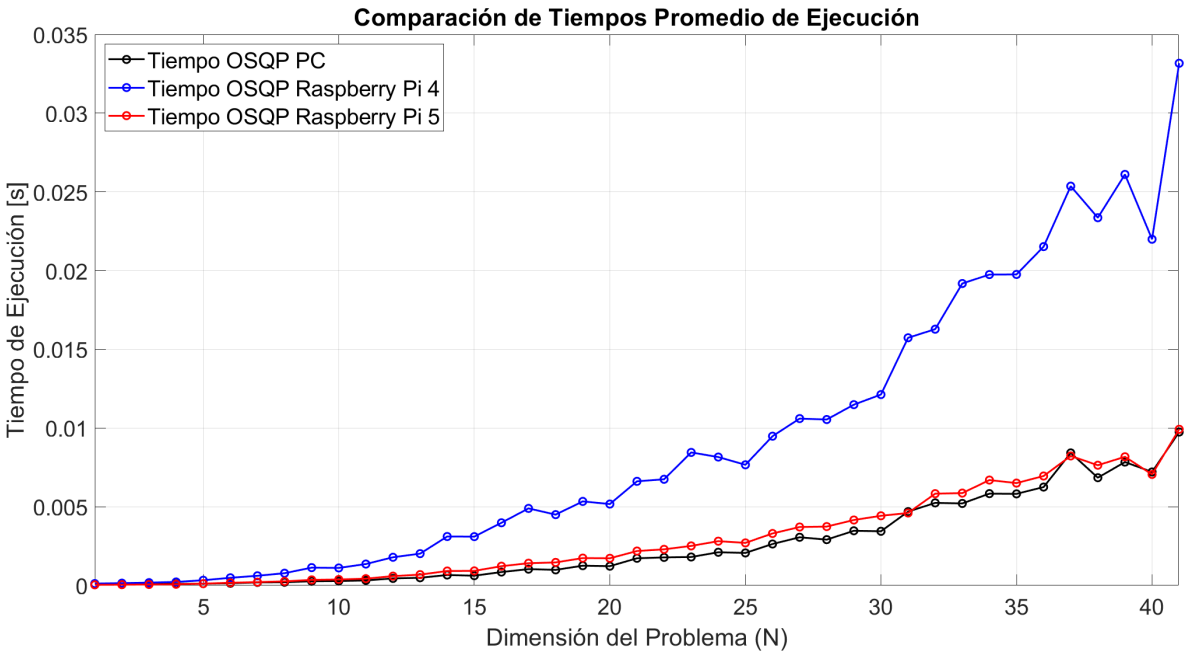

Figura 4.1: Comparación tiempo de ejecución computador, Raspberry Pi 4 y Raspberry Pi 5

por defecto del solver. Posteriormente, se presentan diversas implementaciones que permiten evaluar el comportamiento de `OSQP` en un escenario de control y optimización. Todas las implementaciones empleadas, incluyendo scripts, configuraciones y modelos utilizados se encuentran disponibles en el repositorio de la memoria [11].

Partiendo con la comparación del uso del solver en PC y en las tarjetas Raspberry Pi, en la Tabla 4.1, se comparan los tiempos de ejecución del solver `OSQP` en una Raspberry Pi 4, Raspberry Pi 5 y el computador portátil (para ver la Tabla completa revisar A.13). De esto se tienen las siguientes observaciones:

- **Diferencias en el rendimiento temporal**: En la Raspberry Pi 4, todos los speedups son menores que 1, lo que indica que `OSQP` se ejecuta de forma consistentemente más lenta que en el PC, algo esperable dado el menor poder de cómputo del hardware embebido. Sin embargo, en la Raspberry Pi 5 se observa un comportamiento distinto: para las primeras dimensiones analizadas (de 1×14 a 4×32), así como en un caso aislado (37×230), el rendimiento supera al del computador portátil.
- **La brecha de rendimiento aumenta con la complejidad del problema en Raspberry 4**: Para problemas de menor tamaño (dimensiones entre 1×14 y 3×26), los speedups se sitúan alrededor de 0,5, lo que indica que la Raspberry Pi 4 es aproximadamente el doble de lenta. No obstante, a medida que la dimensión crece (por ejemplo, de 6×44 a 18×116), los speedups caen por debajo de 0,25, lo que implica que la Raspberry Pi 4 puede tardar cuatro veces más o incluso más.
- **Estabilización de rendimiento en dimensiones mayores en Raspberry 4**: A partir de dimensiones superiores a 25×158, los speedups tienden a estabilizarse en el rango de 0,27 a 0,33, con ligeras variaciones. Esto sugiere que, para problemas de mayor escala, el solver mantiene una eficiencia relativa más constante en la Raspberry Pi.

Tabla 4.1: Speedups respecto al computador portátil al ejecutar problemas base en Raspberry Pi 4 y Raspberry Pi 5.

Dimensión NxM	S Raspberry Pi 4	S Raspberry Pi 5
1 x 14	0.540	1.616
2 x 20	0.527	1.572
3 x 26	0.462	1.343
4 x 32	0.448	1.304
5 x 38	0.340	0.990
6 x 44	0.286	0.817
7 x 50	0.314	0.908
8 x 56	0.262	0.780
13 x 86	0.242	0.710
14 x 92	0.213	0.722
16 x 104	0.214	0.695
17 x 110	0.212	0.734
18 x 116	0.220	0.677
25 x 158	0.270	0.764
30 x 188	0.284	0.778
35 x 218	0.294	0.895
37 x 230	0.332	1.024
41 x 254	0.294	0.982

Tabla 4.2: Comparación conjunta de EPP y EMP respecto a `OSQP` en computador portátil ejecutando problemas base en Raspberry Pi.

Dimensión NxM	EPP Raspberry Pi	EMP Raspberry Pi
1x14	0.003	0.080
2x20	1.429e-05	0.001
3x26	1.425e-05	0.014
4x32	0.357	15.298
14x92	0.417	27.330
16x104	0.267	4.314
17x110	0.208	2.704
32x200	0.721	12.372
33x206	0.582	5.482
34x212	0.674	7.845
35x218	0.617	4.029
36x224	0.581	5.434
41x254	0.751	4.140

En términos de la presición de la solución, ambas Raspberry Pi obtienen exactamente los mismos resultados de valor primal, por lo que no se diferencia entre ellas en las tablas de error. A partir de las Tabla de EPP y EMP 4.2 (para ver la Tabla completa revisar A.14 en la sección de Anexos), se tienen las siguientes observaciones sobre la precisión del solver `OSQP` en Raspberry Pi en comparación con el computador portátil:

- En términos generales, el **EPP** se mantiene bajo en la mayoría de los casos, especialmente para problemas de menor dimensión. Para dimensiones hasta aproximadamente 12x80, el error promedio es menor al 0.25 %, lo cual indica que la solución obtenida en Raspberry Pi es bastante cercana a la solución base del computador portátil.
- Sin embargo, a medida que la **dimensión del problema aumenta**, también lo hace el error promedio. A partir de dimensiones superiores a 16x104, se observa una tendencia creciente, alcanzando valores sobre el 0.7 % para las dimensiones más grandes (32x200 y superiores), lo cual podría estar relacionado con limitaciones numéricas o de precisión en la arquitectura ARM del dispositivo.

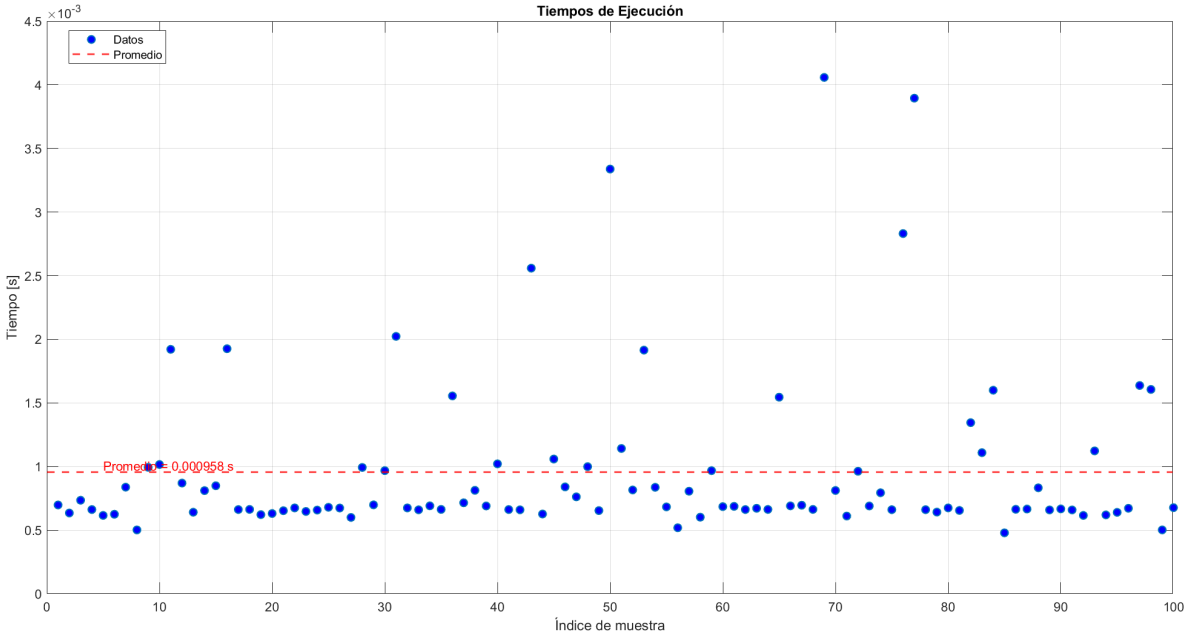

Figura 4.2: Tiempos de ejecución de problemas de dimensión 8×48 en Raspberry Pi 4

- En cuanto al **EMP**, se observan algunos casos aislados con desviaciones significativas, destacando particularmente las dimensiones 4x32 y 14x92, donde el error máximo alcanza valores de 15.30 % y 27.33 % respectivamente. Estos valores atípicos sugieren que para ciertas instancias específicas el solver en Raspberry Pi puede presentar comportamientos numéricos menos estables.
- A pesar de estos peaks, la mayoría de los errores máximos se mantienen bajo el 8 %.

4.2. Comparación de rendimiento entre FPGA y CPU para problema QP de baja dimensión

En estudios previos se ha explorado el uso de aceleradores hardware en FPGA para resolver problemas de optimización cuadrática. Por ejemplo, en [12] se presenta una implementación basada en FPGA que reporta un tiempo de ejecución de aproximadamente 9.2 μs para resolver un problema de dimensión 8×48. Este resultado sirve como línea base de comparación frente a implementaciones sobre CPU tradicionales, como la desarrollada en este trabajo. Se realiza una comparación generarando 100 problemas aleatorios de dimensión 8×48, obteniendo un tiempo promedio de ejecución de 0.000958 s en la Raspberry 4 como se puede ver en la Figura 4.2. Esto implica que la CPU de la Raspberry Pi 4 es aproximadamente 104 veces más lenta que la FPGA. Esto es equivalente a un speedup de 0.0096. Para una comparación más justa entre CPU y la solución altamente optimizada en FPGA, se seleccionó un único problema y se procedió a modificar sus parámetros buscando reducir el tiempo de cómputo lo máximo posible, manteniendo una buena precisión en la solución. Los resultados se presentan en la Tabla 4.3.

Del análisis realizado se observa que:

- La implementación en CPU de la Raspberry 4 ofrece un tiempo de ejecución mayor que

Tabla 4.3: Comparación de tiempos de ejecución y speedup entre FPGA y CPU

FPGA	CPU base	CPU optimizado	S base	S optimizado
$9.2 \cdot 10^{-6}$	$958 \cdot 10^{-6}$	$608 \cdot 10^{-6}$	0.0096	0.01513

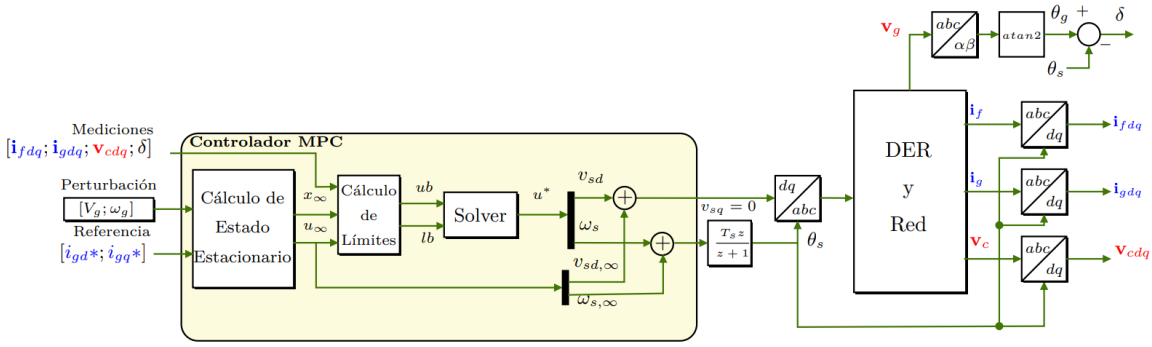

Figura 4.3: Lazo de control DER con un solo controlador

la FPGA, con un speedup de 0.0096 con configuración base.

- La optimización de los parámetros del solver en CPU permite reducir el tiempo de resolución de 958 μs a 608 μs, mejorando el speedup a 0.01513, aunque sigue siendo más lento que la FPGA, puede ser suficiente si la aplicación en la que se está usando el solver requiere un tiempo en el orden de los milisegundos.
- Estos resultados muestran que, para problemas QP de pequeña dimensión, los aceleradores hardware en FPGA ofrecen ventajas en términos de velocidad, mientras que la CPU puede mejorar mediante ajustes de parámetros, aunque sin alcanzar la eficiencia de la FPGA.

4.3. Implementación de problemas QP en tarjetas Raspberry Pi a partir de la simulación de un DER en MATLAB

Siguiendo con las implementaciones que permiten evaluar el comportamiento de `OSQP` en distintos escenarios de control, se presenta una implementación de DER en MATLAB/Simulink obtenida de la Tesis [9], de la cual se extraen los problemas QP para ejecutarse en las tarjetas Raspberry Pi con el objetivo de evaluar los tiempos de ejecución de los problemas QP. La Figura 4.3 muestra el lazo de control considerado, en el cual el bloque encargado de resolver los problemas QP es el denominado 'Solver'. Los problemas obtenidos se transfirieron a ambas plataformas Raspberry Pi, donde se ajustaron los parámetros del solver con el fin de comparar su desempeño.

La simulación incluyó un total de **2501 problemas QP** y estaba configurada con los siguientes parámetros:

- $\alpha = 1{,}1$
- $\rho = 0{,}095$

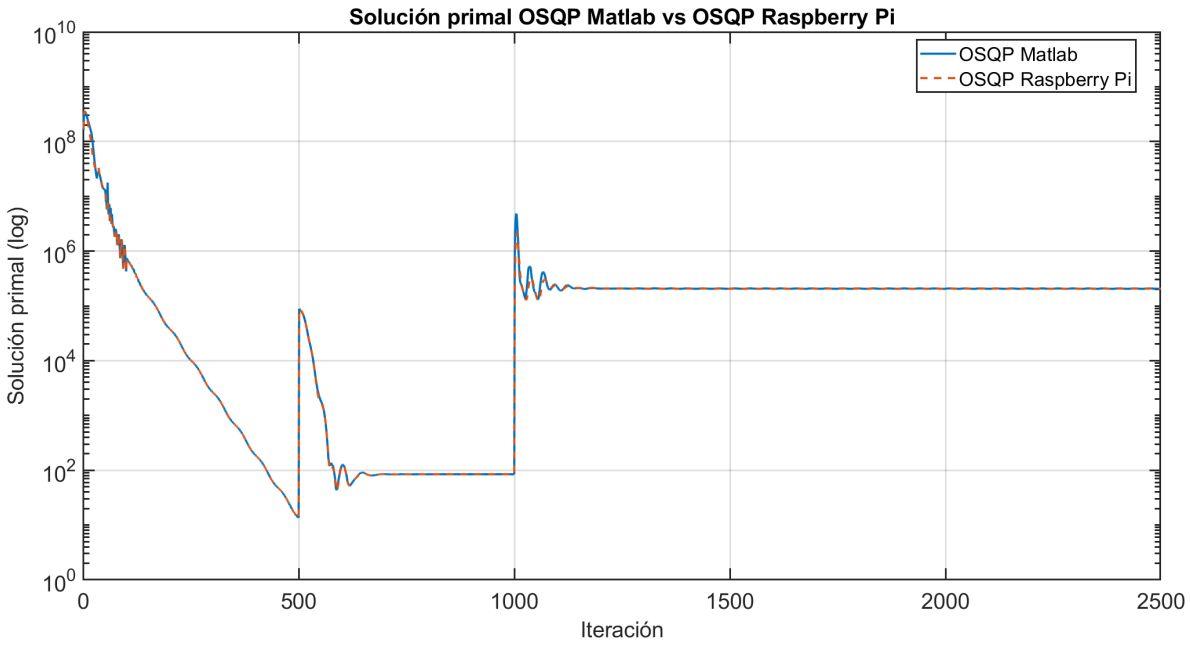

Figura 4.4: Comparación valor primal entre `OSQP` PC y `OSQP` Raspberry Pi

Tabla 4.4: Comparación de tiempos promedio de ejecución entre Raspberry Pi 4 y Raspberry Pi 5.

Dispositivo	**Tiempo promedio** [μs]
Raspberry Pi 4	64.90
Raspberry Pi 5	22.93

- `adaptive_rho` = 0
- `scaling` = 1000
- `max_iter` = 30
- `check_termination` = 5
- Demás parámetros por defecto

En las pruebas realizadas se observa que los parámetros seleccionados inicialmente resultan bastante eficientes. Para verificar su comportamiento, se efectuaron experimentos modificando el valor de α a 1.8, 1.6 y 0.8. Dado que es sabido que el parámetro *warm_start* influye en el tiempo de ejecución, se decidió desactivarlo con el fin de evaluar su impacto. Asimismo, se llevó a cabo una prueba desactivando *scaling*, otra activando *adaptive_rho* y, finalmente, una prueba combinando la activación de *adaptive_rho* con un cambio en el valor de eps_rel, pasando de 1e-3 a 1e-2. Los tiempos de cómputo promedio obtenidos para ambas Raspberry se muestran en la Tabla 4.4. En dichos resultados se aprecia que la Raspberry Pi 5 alcanza tiempos menores que la Raspberry Pi 4.

En términos de exactitud numérica, no se observaron diferencias en el valor primal entre

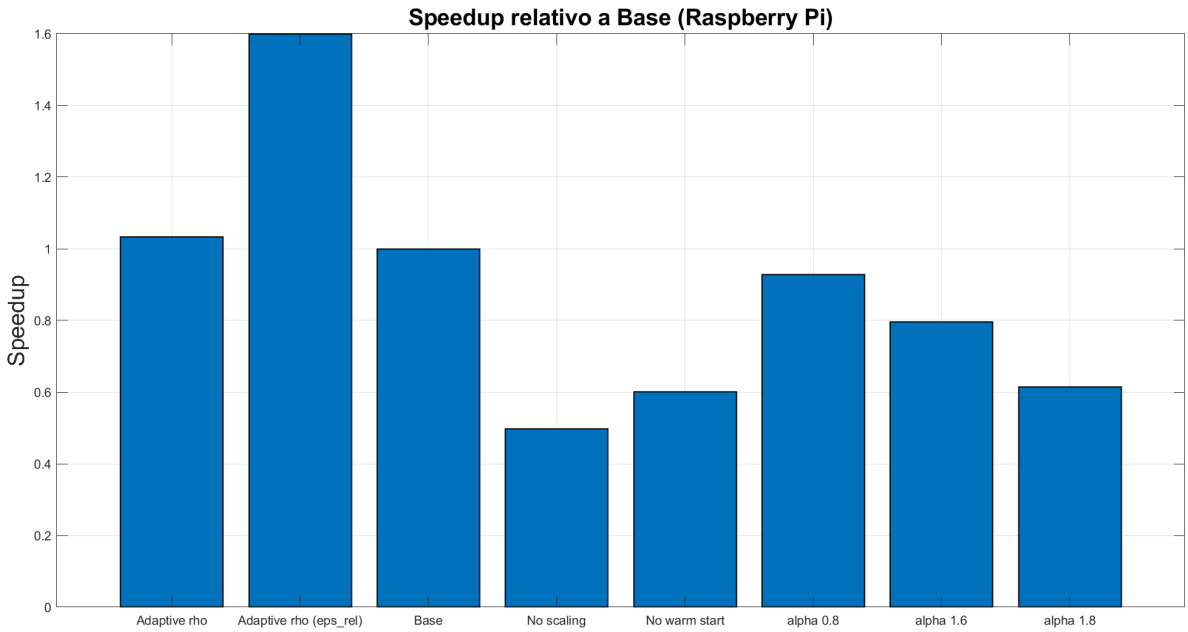

Figura 4.5: Speedup Raspberry Pi respecto a configuración por defecto

ambas Raspberry Pi. Asimismo, la comparación con los resultados obtenidos en PC mostró un **EPP del 1.6 %**, como se ilustra en la Figura 4.4. Por su parte, la Figura 4.5 presenta la comparación de speedup bajo diferentes configuraciones del solver, donde ambas Raspberry Pi mantienen el mismo comportamiento al variar los parámetros.

Observaciones

Del conjunto de pruebas realizadas se observa que:

- La precisión del solver `OSQP` es consistente en ambas Raspberry Pi, obteniendo los mismos valores primales entre ellas y que además son equivalentes con los obtenidos en PC.
- El parámetro warm_start representa aproximadamente un 40 % del tiempo de la solución.
- Desactivar o seleccionar de manera inadecuada el parámetro scaling puede impedir o dificultar la convergencia del problema QP.
- El rendimiento computacional sí presenta diferencias relevantes: la Raspberry Pi 5 es, en promedio, **tres veces más rápida** que la Raspberry Pi 4, manteniendo las mismas tendencias frente a modificaciones en los parámetros del solver.
- Si se requieren tiempos de ejecución en el orden los los micro segundos, ambas plataformas son adecuadas para ejecutar `OSQP`, pero la Raspberry Pi 5 ofrece una ventaja significativa en tiempos de cómputo, especialmente relevante para aplicaciones en tiempo real.

4.4. Reemplazo de solver `quadprog` por `OSQP` en lazo de control en cascada de un sistema DER

Se tiene un lazo de control en cascada implementado en MATLAB/Simulink obtenido de la Tesis [10] que utiliza inicialmente el solver `quadprog`. En el computador portátil, se

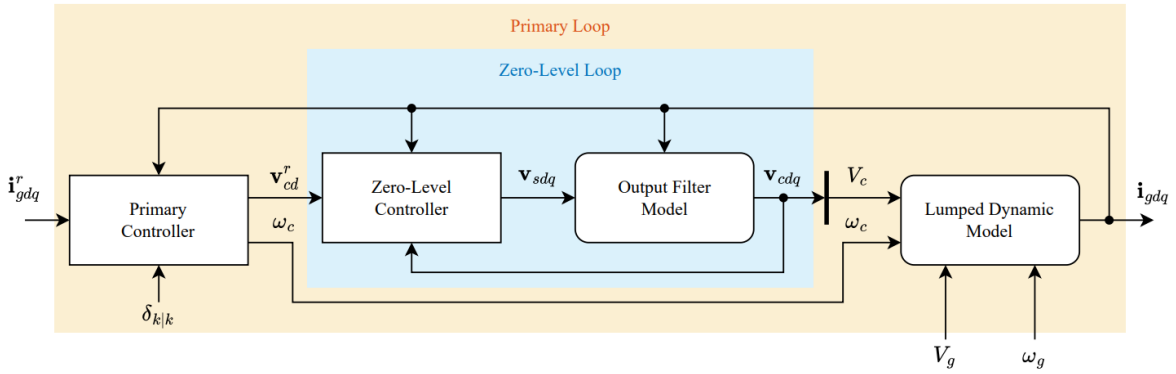

Figura 4.6: Lazo de control en cascada DER con dos controladores

Tabla 4.5: Speedup respecto a `OSQP` base y EPP respecto a `quadprog` de problemas QP de lazo de control en cascada.

Controlador	**Speedup**	**EPP respecto a `quadprog`**
Zero-Level Controller	1.085	1.278e-01
Primary Controller	1.137	2.815e-04

reemplaza el solver `quadprog` por `OSQP` con el fin de verificar la equivalencia funcional entre las soluciones obtenidas por ambos con parámetros por defecto. La Figura 4.6 muestra el lazo de control considerado, en el cual se resuelven problemas QP tanto en el bloque 'Primary Controller' como en el 'Zero-Level Controller'. Por ello, el reemplazo de `quadprog` por `OSQP` se realiza en ambos bloques.

En la Figura 4.7 se observan los valores primales obtenidos en el control de orden zero, donde la comparación entre `OSQP` y `quadprog` muestra un error promedio de 0.126 %. En la Figura 4.8 se presentan los valores primales del control primario, con un error promedio de $4 \cdot 10^{-4}$ %. A continuación, se modificaron parámetros con el fin de reducir los tiempos de ejecución, procurando mantener la equivalencia de los valores primales. Si bien se realizaron diversas pruebas ajustando los parámetros de ambos controladores, la sintonización orientada específicamente a disminuir dichos tiempos resultó compleja. Esto se debe a que, al optimizar los parámetros del solver de un controlador, el tiempo de ejecución del solver del otro podía incrementarse. Por este motivo, se optó por ajustar únicamente el parámetro *eps_rel*, fijándolo en 1e-2. Con este cambio, fue posible reducir de manera simultánea los tiempos de ejecución en ambos controladores, obteniendo un speedup de 1.085 para el Zero-Level Controller y de 1.137 para el Primary Controller. Los resultados correspondientes a esta nueva configuración se presentan en la Tabla 4.5.

Observaciones

A partir de los resultados obtenidos se observa que:

- Existe una equivalencia funcional entre los solvers `quadprog` y `OSQP` en el contexto del lazo de control en cascada evaluado.
- Los errores promedio bajos (0.126 % en zero-level control y $4 \cdot 10^{-4}$ % en primary

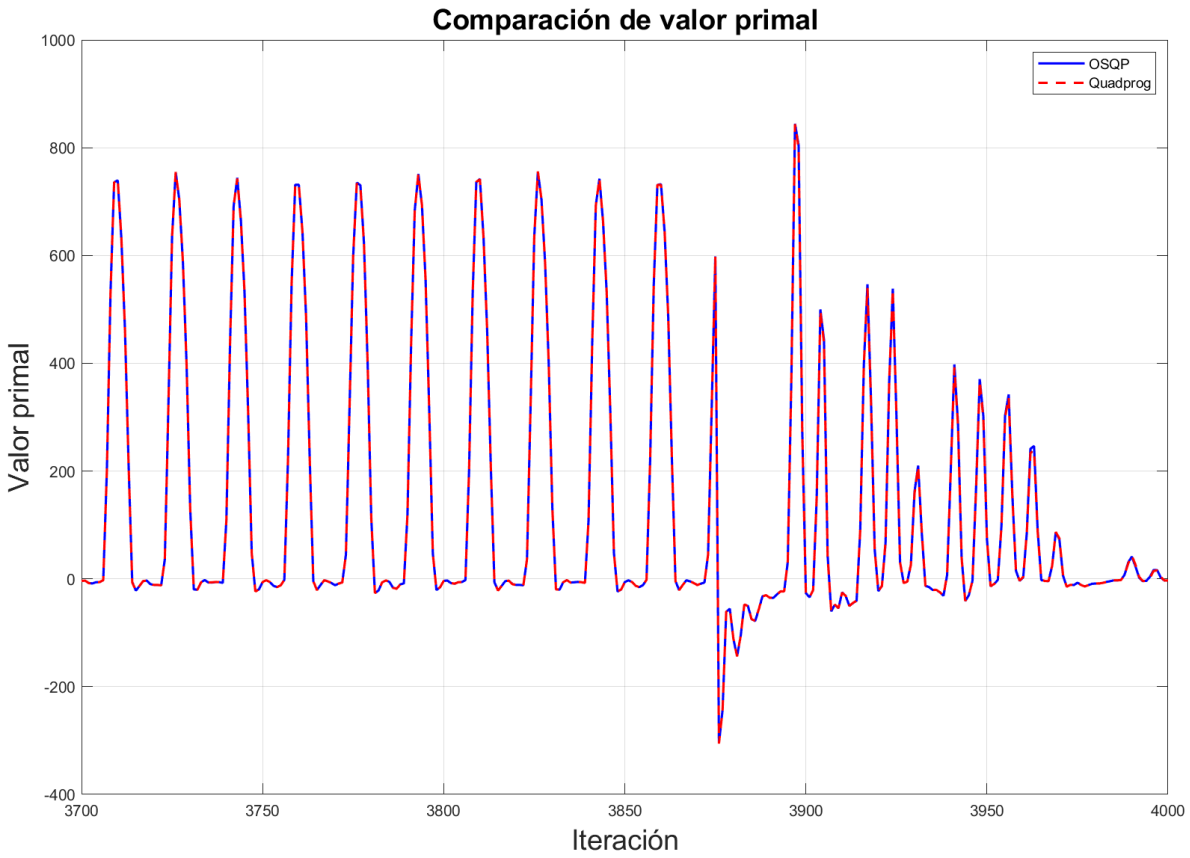

Figura 4.7: Valor primal zero-level control

control) con parámetros por defecto, confirman que ambos solvers producen soluciones prácticamente idénticas.

- Es posible la intercambiabilidad de `OSQP` y `quadprog` dentro de un lazo de control diseñado originalmente para uno de ellos, manteniendo la funcionalidad y desempeño esperado.
- En un lazo de control con dos controladores, la modificación de parámetros en los solvers en cada controlador puede reducir el tiempo de ejecución de uno de los controladores, pero puede incrementar el tiempo de ejecución del otro, complicando la sintonización en presencia de dos controladores. Por esta razón, se optó por modificar únicamente el parámetro *eps_rel*, buscando mejorar el rendimiento de ambos controladores, sin afectar la estabilidad o la precisión de los resultados.

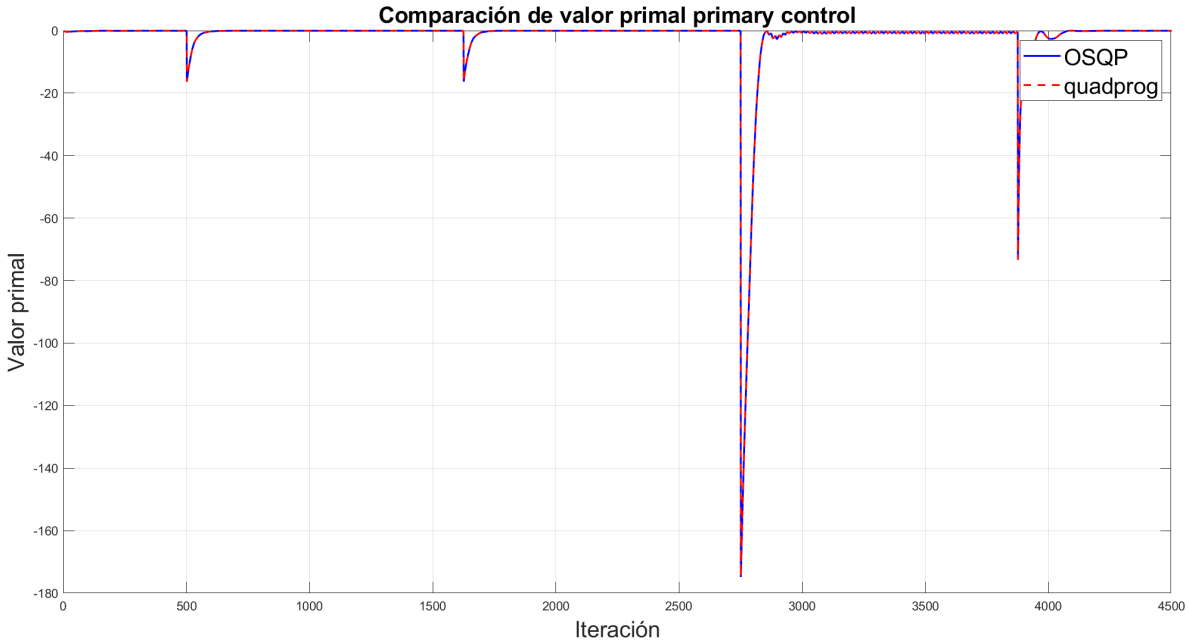

Figura 4.8: Valor primal primary control

5 | Conclusiones y trabajo futuro

A lo largo del desarrollo de esta memoria se llevó a cabo una caracterización profunda del solver `OSQP` en la resolución de problemas QP, típicamente asociados a algoritmos de MPC. El estudio se abordó desde una perspectiva práctica, abarcando pruebas en plataformas de propósito general y embebidas, análisis paramétricos, validación cruzada con `quadprog` y evaluación en lazos de control reales. En conjunto, estos elementos permitieron obtener una visión más amplia del comportamiento del solver bajo condiciones diversas y representativas de aplicaciones reales.

Los resultados obtenidos sugieren que `OSQP` es un solver eficiente y altamente configurable, capaz de resolver problemas QP con una relación costo-beneficio favorable en términos de precisión, velocidad y portabilidad. Esta combinación lo convierte en un buen candidato para su integración en aplicaciones de control en tiempo real, especialmente en sistemas embebidos con restricciones de procesamiento. En este sentido, se comprobó que la configuración de parámetros influye decisivamente en el rendimiento. Valores como α, ρ, eps_rel y σ afectan directamente la velocidad de convergencia, el número de iteraciones y la estabilidad numérica. Una sintonización adecuada de estos parámetros puede reducir significativamente los tiempos de ejecución, incluso en problemas de mediana a gran escala, lo que refuerza la importancia de una selección informada en contextos prácticos. La comparación con `quadprog` complementó este análisis al demostrar que `OSQP` mantiene errores primales relativamente bajos bajo condiciones equivalentes, lo que respalda su confiabilidad desde el punto de vista numérico.

Asimismo, las pruebas realizadas en plataformas Raspberry Pi 4 y 5 confirmaron la viabilidad del despliegue embebido, especialmente al emplear configuraciones ajustadas del solver. Las mejoras observadas en el tiempo de ejecución con respecto a la configuración por defecto evidencian que una sintonización adecuada puede aportar beneficios significativos en el rendimiento del sistema, facilitando la implementación en entornos con recursos computacionales limitados. La integración del solver en lazos de control reales reforzó esta conclusión al demostrar su aplicabilidad funcional, incluso al comparar la ejecución en MATLAB con su contraparte en C sobre hardware embebido. Este conjunto de resultados pone de manifiesto la portabilidad del solver y abre la puerta a implementaciones en escenarios de control más exigentes, donde es necesario equilibrar precisión, tiempos de respuesta y cargas de cómputo.

En términos generales, este trabajo ofrece una guía práctica para facilitar la elección y el ajuste de los parámetros del solver en aplicaciones reales, y constituye un punto de partida para seguir explorando el uso de `OSQP` en sistemas de control embebidos. Como línea de continuidad, se propone comparar `OSQP` con otros solvers de problemas QP a fin de evaluar diferencias de desempeño, consumo de memoria y facilidad de integración en distintos tipos de proyectos. En este sentido, una alternativa interesante es ODYS [20]. Aunque no formaba parte del conjunto

de herramientas consideradas inicialmente, su relevancia se identificó durante el desarrollo del trabajo, ya que, pese a ser un solver de licencia comercial, está específicamente orientado a aplicaciones de MPC, cuenta con una implementación eficiente en C y recibe actualizaciones continuas. También sería útil estudiar cómo se comporta el solver en dispositivos más limitados que las plataformas utilizadas en este trabajo, como microcontroladores ARM Cortex-M, para identificar en qué momento es necesario recurrir a hardware más potente o a aceleradores especializados.

Otra línea consiste en automatizar el ajuste de parámetros mediante técnicas como búsqueda en malla o aprendizaje por refuerzo. Estas herramientas permitirían adaptar el funcionamiento del solver según las necesidades del problema o del dispositivo donde se ejecute, algo especialmente útil en aplicaciones que cambian con el tiempo. En este mismo sentido, el desarrollo de un asistente que sugiera valores de parámetros adecuados según la complejidad del problema, la plataforma utilizada y los requisitos de velocidad facilitaría mucho su uso en proyectos reales.

En conclusión, este trabajo no sólo demuestra que `OSQP` es adecuado para aplicaciones de control predictivo en sistemas embebidos, sino que también plantea un camino para seguir mejorando y extendiendo su uso. Al conectar el análisis numérico con las limitaciones reales del hardware, se contribuye al desarrollo de soluciones de control más eficientes, accesibles y fáciles de reproducir. Finalmente, la existencia de un repositorio completo y documentado permite repetir, ampliar y adaptar los experimentos realizados, asegurando que este trabajo pueda servir de apoyo a futuras investigaciones. Se dejan disponibles los códigos utilizados en el repositorio correspondiente [11].

A | Anexos

A.1. Comparación inicial

Tabla A.1: Tiempos promedio, desviación estándar y speedup entre `quadprog` y `OSQP` por dimensión del problema.

Dimensión NxM	Tiempo `quadprog` [µs]	Desv. std `quadprog`	Tiempo `OSQP` [µs]	Desv. std `OSQP`	Speedup
1 x 14	1448.6	168.41	64.28	3.9031	22.536
2 x 20	1503.6	176.17	77.23	34.033	19.469
3 x 26	1496.4	142.45	82.81	22.167	18.070
4 x 32	1466.7	132.64	99.86	33.779	14.688
5 x 38	1519.2	140.27	112.49	26.936	13.506
6 x 44	1603.3	150.80	139.28	54.896	11.511
7 x 50	1646.0	138.35	193.11	180.77	8.5238
8 x 56	1708.2	142.62	204.16	114.09	8.3672
9 x 62	1792.8	128.49	266.77	203.55	6.7203
10 x 68	1817.7	145.26	281.47	182.22	6.4577
11 x 74	1865.5	150.35	318.79	231.44	5.8518
12 x 80	1870.1	135.84	446.92	501.84	4.1844
13 x 86	2269.0	803.06	488.62	404.25	4.6438
14 x 92	2241.6	527.92	662.93	732.37	3.3814
15 x 98	2626.9	637.41	619.22	553.44	4.2423
16 x 104	2869.4	832.42	850.66	839.09	3.3732
17 x 110	3009.6	566.62	1037.5	1294.7	2.9007
18 x 116	3149.1	789.95	990.57	1175.8	3.1791
19 x 122	3435.8	826.42	1255.0	1461.1	2.7376
20 x 128	3424.6	795.73	1225.5	1294.4	2.7945
21 x 134	3927.6	683.80	1726.4	2102.2	2.2751
22 x 140	4018.1	752.80	1782.2	1921.9	2.2545
23 x 146	4312.8	894.02	1808.2	2255.3	2.3851
24 x 152	4175.2	761.88	2108.2	2436.5	1.9804
25 x 158	4697.9	752.59	2066.4	2108.1	2.2734
26 x 164	5000.7	858.96	2632.4	3322.6	1.8997
27 x 170	5438.9	978.66	3060.6	3800.1	1.7771
28 x 176	5407.7	902.80	2909.8	3627.4	1.8585
29 x 182	6156.2	1093.8	3473.8	4864.1	1.7722
30 x 188	6442.6	1369.8	3441.3	4001.4	1.8722
31 x 194	6962.4	1196.5	4689.3	5307.9	1.4847
32 x 200	6897.8	1325.4	5244.5	6906.4	1.3152
33 x 206	8120.0	1949.1	5208.8	5925.1	1.5589
34 x 212	8530.6	1778.5	5832.8	6611.4	1.4625
35 x 218	8950.4	1877.7	5817.4	7092.6	1.5386
36 x 224	8884.4	1777.6	6249.4	7367.0	1.4216
37 x 230	10337.0	2032.0	8419.1	8835.8	1.2277
38 x 236	10554.0	2309.6	6844.5	7953.5	1.5420
39 x 242	14427.0	3272.7	7837.6	9555.8	1.8408
40 x 248	13716.0	4513.3	7208.4	9262.0	1.9027
41 x 254	16009.0	4253.1	9744.8	11163.0	1.6428

Tabla A.2: Errores absolutos y porcentuales de la solución primal de `OSQP` respecto a `quadprog`.

Dimensión NxM	**Error abs. promedio**	**EPP**	**Error máx.**	**EMP**
1 x 14	0.000	0.012	0.001	0.095
2 x 20	0.002	0.016	0.018	0.156
3 x 26	0.005	0.017	0.024	0.080
4 x 32	0.060	0.106	4.277	7.510
5 x 38	0.038	0.034	0.266	0.237
6 x 44	0.078	0.050	0.957	0.614
7 x 50	0.155	0.065	0.959	0.405
8 x 56	0.191	0.062	1.064	0.345
9 x 62	0.379	0.087	1.929	0.444
10 x 68	0.402	0.078	1.737	0.339
11 x 74	0.503	0.081	2.750	0.440
12 x 80	0.736	0.100	3.490	0.473
13 x 86	0.951	0.107	3.259	0.366
14 x 92	1.352	0.128	32.081	3.034
15 x 98	1.274	0.111	11.412	0.991
16 x 104	1.822	0.122	6.625	0.445
17 x 110	1.772	0.116	5.941	0.390
18 x 116	2.588	0.146	41.414	2.334
19 x 122	2.789	0.128	12.459	0.571
20 x 128	2.847	0.123	7.781	0.336
21 x 134	3.477	0.136	17.443	0.681
22 x 140	3.874	0.139	9.712	0.350
23 x 146	4.903	0.160	77.864	2.541
24 x 152	5.471	0.163	75.405	2.241
25 x 158	5.781	0.146	16.934	0.428
26 x 164	5.606	0.143	16.244	0.415
27 x 170	6.316	0.145	23.772	0.546
28 x 176	9.043	0.202	231.230	5.165
29 x 182	9.610	0.200	193.610	4.032
30 x 188	9.868	0.185	102.990	1.935
31 x 194	11.392	0.193	276.890	4.682
32 x 200	10.597	0.170	74.705	1.202
33 x 206	16.735	0.247	477.130	7.039
34 x 212	12.553	0.176	108.070	1.515
35 x 218	12.624	0.166	94.699	1.248
36 x 224	16.106	0.202	192.080	2.404
37 x 230	16.782	0.201	186.270	2.231
38 x 236	21.193	0.238	602.960	6.783
39 x 242	25.835	0.272	403.130	4.246
40 x 248	17.692	0.190	161.910	1.742
41 x 254	23.540	0.228	324.190	3.134

A.2. Parámetro α

Tabla A.3: Speedups respecto a `OSQP` base para distintos valores del parámetro α.

Dimensión NxM	**S** $\alpha = 0.2$	**S** $\alpha = 0.4$	**S** $\alpha = 1$	**S** $\alpha = 1.8$
1 x 14	0.775	0.825	0.992	0.885
2 x 20	0.774	0.924	0.987	0.932
3 x 26	0.722	0.879	1.017	0.854
4 x 32	0.472	0.703	0.870	0.809
5 x 38	0.407	0.598	0.867	0.808
6 x 44	0.311	0.515	0.868	0.859
7 x 50	0.267	0.478	0.915	0.992
8 x 56	0.240	0.427	0.813	0.933
9 x 62	0.216	0.404	0.749	0.881
10 x 68	0.207	0.398	0.786	0.945
11 x 74	0.210	0.376	0.756	0.926
12 x 80	0.218	0.385	0.789	1.057
13 x 86	0.194	0.353	0.695	0.994
14 x 92	0.209	0.378	0.768	1.009
15 x 98	0.200	0.383	0.758	1.000
16 x 104	0.192	0.339	0.687	0.987
17 x 110	0.227	0.385	0.700	1.015
18 x 116	0.209	0.348	0.712	0.963
19 x 122	0.220	0.350	0.697	0.966
20 x 128	0.198	0.355	0.738	1.055
21 x 134	0.237	0.379	0.690	0.952
22 x 140	0.227	0.372	0.703	1.026
23 x 146	0.222	0.377	0.721	1.025
24 x 152	0.225	0.375	0.775	1.079
25 x 158	0.213	0.350	0.699	1.038
26 x 164	0.250	0.416	0.772	1.033
27 x 170	0.288	0.436	0.758	1.163
28 x 176	0.263	0.373	0.696	1.029
29 x 182	0.301	0.466	0.786	1.040
30 x 188	0.255	0.387	0.689	1.033
31 x 194	0.279	0.446	0.809	1.065
32 x 200	0.332	0.424	0.765	1.108
33 x 206	0.294	0.442	0.740	1.058
34 x 212	0.309	0.443	0.738	1.061
35 x 218	0.317	0.456	0.766	1.070
36 x 224	0.307	0.451	0.746	1.084
37 x 230	0.346	0.472	0.777	1.075
38 x 236	0.293	0.454	0.762	1.039
39 x 242	0.320	0.461	0.777	1.085
40 x 248	0.322	0.457	0.794	1.081
41 x 254	0.349	0.489	0.770	1.040

Tabla A.4: Comparación conjunta de EPP y EMP de la solución primal según el parámetro α en OSQP.

	$\alpha = 0.2$		$\alpha = 0.4$		$\alpha = 1$		$\alpha = 1.8$	
Dimensión NxM	**EPP**	**EMP**	**EPP**	**EMP**	**EPP**	**EMP**	**EPP**	**EMP**
1 x 14	0.013	0.098	0.012	0.096	0.012	0.095	0.015	0.095
2 x 20	0.016	0.159	0.016	0.157	0.016	0.156	0.021	0.158
3 x 26	0.020	0.135	0.019	0.149	0.017	0.080	0.019	0.084
4 x 32	0.038	0.312	0.033	0.403	0.021	0.250	0.023	0.224
5 x 38	0.049	0.751	0.685	65.470	0.024	0.327	0.023	0.254
6 x 44	0.070	1.902	0.044	0.841	0.032	0.489	0.047	0.828
7 x 50	0.247	17.973	0.475	42.710	0.037	0.508	0.050	0.582
8 x 56	0.103	3.700	0.046	0.384	0.034	0.344	0.038	0.304
9 x 62	0.457	20.902	0.084	3.866	0.037	0.368	0.038	0.276
10 x 68	0.082	2.588	0.036	0.228	0.036	0.428	0.037	0.219
11 x 74	0.076	0.946	0.032	0.145	0.027	0.176	0.027	0.247
12 x 80	0.451	27.283	0.046	0.780	0.043	0.642	0.043	0.966
13 x 86	0.325	13.729	0.045	0.921	0.028	0.277	0.028	0.537
14 x 92	2.865	263.290	0.935	89.131	0.301	27.285	0.048	2.077
15 x 98	0.261	8.922	0.037	0.335	0.030	0.245	0.032	0.273
16 x 104	0.643	12.747	0.078	1.924	0.026	0.335	0.032	0.313
17 x 110	0.512	11.265	0.143	6.117	0.044	1.406	0.030	0.218
18 x 116	0.680	24.139	0.261	20.419	0.122	9.816	0.033	0.425
19 x 122	0.855	12.683	0.131	3.665	0.032	0.641	0.031	0.328
20 x 128	0.388	8.844	0.068	2.790	0.037	0.686	0.035	0.220
21 x 134	0.689	12.573	0.203	6.260	0.043	0.824	0.040	0.647
22 x 140	0.552	12.504	0.094	2.675	0.031	0.277	0.028	0.170
23 x 146	0.883	28.463	0.331	21.649	0.139	9.844	0.088	5.733
24 x 152	1.046	30.553	0.259	10.978	0.082	5.245	0.065	3.962
25 x 158	0.536	24.000	0.062	2.087	0.032	0.230	0.032	0.173
26 x 164	0.806	12.379	0.200	5.229	0.065	2.171	0.030	0.252
27 x 170	0.917	15.423	0.232	8.407	0.052	1.023	0.032	0.431
28 x 176	1.045	15.409	0.255	5.042	0.069	1.822	0.036	0.506
29 x 182	1.459	35.142	0.488	10.779	0.127	4.428	0.034	0.384
30 x 188	1.091	26.214	0.419	16.549	0.098	4.432	0.039	0.736
31 x 194	1.015	20.033	0.160	2.713	0.042	0.877	0.028	0.369
32 x 200	2.087	32.536	0.662	18.653	0.074	1.385	0.043	0.730
33 x 206	1.412	35.859	0.384	13.309	0.089	4.446	0.041	0.861
34 x 212	1.876	39.675	0.443	7.153	0.095	3.112	0.028	0.338
35 x 218	1.622	26.099	0.496	12.211	0.069	2.395	0.023	0.367
36 x 224	1.404	25.714	0.377	8.783	0.080	1.977	0.036	0.608
37 x 230	1.497	15.658	0.432	4.895	0.097	2.106	0.027	0.464
38 x 236	1.311	36.868	0.327	11.200	0.095	3.456	0.027	0.430
39 x 242	1.458	24.317	0.580	11.188	0.166	6.135	0.055	1.657
40 x 248	1.000	12.701	0.342	6.768	0.110	2.749	0.092	2.646
41 x 254	1.578	15.197	0.512	6.585	0.125	1.693	0.046	1.148

A.3. Parámetro eps_rel

Tabla A.5: Speedups para distintos valores del parámetro `eps_rel` en `OSQP`.

Dimensión NxM	eps_rel 1e-5	eps_rel 1e-4	eps_rel 1e-2	eps_rel 1e-1
1 x 14	0.971	0.963	0.988	0.863
2 x 20	0.969	0.927	1.018	0.852
3 x 26	1.039	0.944	0.994	0.890
4 x 32	0.888	0.901	1.046	0.976
5 x 38	0.755	0.862	1.055	0.972
6 x 44	0.674	0.850	1.109	1.095
7 x 50	0.652	0.898	1.314	1.352
8 x 56	0.557	0.795	1.240	1.296
9 x 62	0.485	0.781	1.294	1.499
10 x 68	0.549	0.790	1.279	1.427
11 x 74	0.487	0.784	1.387	1.312
12 x 80	0.437	0.784	1.497	1.695
13 x 86	0.390	0.724	1.496	1.691
14 x 92	0.427	0.724	1.610	1.674
15 x 98	0.403	0.640	1.511	1.943
16 x 104	0.402	0.683	1.614	2.635
17 x 110	0.447	0.690	1.838	2.911
18 x 116	0.394	0.622	1.546	2.645
19 x 122	0.405	0.702	1.732	3.146
20 x 128	0.403	0.701	1.719	2.713
21 x 134	0.435	0.699	1.904	3.533
22 x 140	0.422	0.642	1.826	3.367
23 x 146	0.428	0.696	1.646	3.122
24 x 152	0.399	0.714	1.616	3.214
25 x 158	0.361	0.629	1.829	2.965
26 x 164	0.442	0.725	1.926	3.688
27 x 170	0.456	0.682	2.033	4.363
28 x 176	0.452	0.722	1.773	3.932
29 x 182	0.476	0.752	1.846	4.139
30 x 188	0.424	0.661	1.788	3.806
31 x 194	0.480	0.769	2.522	4.950
32 x 200	0.535	0.824	2.116	5.747
33 x 206	0.460	0.617	2.082	4.854
34 x 212	0.479	0.624	1.848	5.093
35 x 218	0.499	0.641	1.844	5.009
36 x 224	0.468	0.655	1.889	4.961
37 x 230	0.490	0.651	2.141	6.435
38 x 236	0.477	0.657	2.017	5.003
39 x 242	0.474	0.679	1.926	5.387
40 x 248	0.489	0.670	1.990	4.709
41 x 254	0.492	0.687	1.839	5.930

Tabla A.6: EPP y EMP para distintos valores de `eps_rel` en `OSQP`.

	eps_rel = 1e-5		**eps_rel = 1e-4**		**eps_rel = 1e-2**		**eps_rel = 1e-1**	
Dimensión NxM	**EPP**	**EMP**	**EPP**	**EMP**	**EPP**	**EMP**	**EPP**	**EMP**
1 x 14	0.012	0.095	0.011	0.095	0.009	0.097	0.009	0.097
2 x 20	0.016	0.156	0.015	0.156	0.001	0.041	0.001	0.041
3 x 26	0.017	0.080	0.017	0.080	0.002	0.067	0.002	0.067
4 x 32	0.030	0.297	0.028	0.272	0.156	3.802	1.014	40.234
5 x 38	0.033	0.233	0.030	0.215	0.374	14.009	3.049	36.766
6 x 44	0.050	0.608	0.045	0.555	0.426	5.649	4.880	106.990
7 x 50	0.064	0.401	0.059	0.366	0.610	7.877	6.012	93.259
8 x 56	0.061	0.343	0.056	0.309	0.708	5.899	5.588	55.501
9 x 62	0.084	0.440	0.078	0.399	0.852	4.678	7.466	95.955
10 x 68	0.077	0.336	0.072	0.316	0.731	3.315	7.835	80.261
11 x 74	0.079	0.436	0.072	0.397	0.715	3.871	5.044	51.093
12 x 80	0.098	0.468	0.090	0.426	0.951	8.732	8.201	89.270
13 x 86	0.106	0.363	0.096	0.336	0.999	3.895	7.676	54.659
14 x 92	0.369	27.285	0.088	0.476	1.112	6.642	7.485	97.016
15 x 98	0.191	9.218	0.104	1.313	1.014	5.156	7.291	52.790
16 x 104	0.119	0.441	0.110	0.397	1.228	5.048	10.329	56.798
17 x 110	0.124	1.406	0.102	0.354	1.187	3.661	7.678	43.798
18 x 116	0.217	9.816	0.111	0.558	1.118	3.504	9.250	85.416
19 x 122	0.125	0.641	0.109	0.351	1.288	8.944	9.781	62.986
20 x 128	0.122	0.364	0.111	0.303	1.269	4.344	8.327	65.695
21 x 134	0.129	0.824	0.113	0.326	1.253	4.329	9.516	54.257
22 x 140	0.132	0.338	0.123	0.316	1.334	4.246	9.179	49.188
23 x 146	0.240	9.844	0.118	0.335	1.239	4.535	9.381	62.867
24 x 152	0.190	5.245	0.123	0.327	1.298	3.247	10.516	103.340
25 x 158	0.141	0.424	0.130	0.386	1.377	3.743	10.109	71.432
26 x 164	0.170	2.171	0.125	0.375	1.385	6.075	8.943	50.864
27 x 170	0.149	1.023	0.120	0.455	1.392	7.558	9.098	37.318
28 x 176	0.178	1.822	0.125	0.273	1.309	6.336	9.284	48.582
29 x 182	0.225	4.428	0.116	0.524	1.296	4.676	9.360	77.552
30 x 188	0.203	4.432	0.131	0.326	1.480	5.812	8.830	48.501
31 x 194	0.144	0.877	0.123	0.367	1.471	5.512	10.438	42.590
32 x 200	0.178	1.385	0.125	0.324	1.477	5.729	11.912	80.335
33 x 206	0.203	4.446	0.137	0.390	1.506	5.136	12.178	68.919
34 x 212	0.197	3.112	0.130	0.532	1.582	8.097	11.099	84.842
35 x 218	0.171	2.395	0.125	0.327	1.416	4.424	11.134	68.700
36 x 224	0.190	1.977	0.126	0.362	1.501	7.365	10.276	45.920
37 x 230	0.213	2.106	0.137	0.320	1.570	4.211	11.990	38.763
38 x 236	0.205	3.456	0.126	0.286	1.489	4.489	10.286	58.847
39 x 242	0.275	6.135	0.131	0.434	1.483	4.933	11.285	72.028
40 x 248	0.211	2.749	0.126	0.315	1.369	5.579	10.083	54.414
41 x 254	0.231	1.693	0.129	0.332	1.503	5.222	11.415	44.347

A.4. Parámetro σ

Tabla A.7: Speedups para distintos valores del parámetro σ en `OSQP`.

Dimensión NxM	σ **= 1e-10**	σ **= 1e-8**	σ **= 1e-4**	σ **= 1e-2**
1 x 14	1.336	0.991	0.981	0.990
2 x 20	1.292	1.019	0.990	0.980
3 x 26	1.249	1.028	0.971	1.024
4 x 32	1.264	0.980	0.998	1.000
5 x 38	1.143	0.936	0.941	0.949
6 x 44	1.040	0.997	0.984	0.971
7 x 50	1.115	1.071	1.088	1.100
8 x 56	0.959	0.978	0.964	0.989
9 x 62	0.958	0.946	0.974	0.951
10 x 68	0.999	1.022	1.003	1.017
11 x 74	0.851	0.993	1.004	1.035
12 x 80	1.063	1.058	1.071	1.071
13 x 86	1.011	0.980	1.002	0.995
14 x 92	0.987	0.966	0.992	1.003
15 x 98	1.024	1.015	1.033	1.039
16 x 104	0.956	0.984	0.988	0.986
17 x 110	0.983	0.985	0.984	0.983
18 x 116	0.978	0.992	0.985	0.966
19 x 122	0.969	0.984	0.988	0.968
20 x 128	0.974	1.018	1.013	1.013
21 x 134	1.003	0.998	0.976	0.989
22 x 140	0.993	0.983	0.998	0.983
23 x 146	1.003	1.002	1.002	1.005
24 x 152	0.976	0.995	1.000	0.981
25 x 158	0.999	0.989	0.993	1.004
26 x 164	0.996	1.013	1.018	1.003
27 x 170	1.030	1.011	1.030	1.024
28 x 176	1.006	1.012	0.973	0.999
29 x 182	0.944	0.980	0.950	0.954
30 x 188	0.983	0.987	0.994	0.974
31 x 194	1.006	1.016	1.015	1.021
32 x 200	1.126	1.142	1.165	1.126
33 x 206	0.976	1.003	0.982	0.999
34 x 212	0.973	0.974	0.967	0.972
35 x 218	0.992	0.993	0.994	1.006
36 x 224	0.999	0.992	1.014	1.012
37 x 230	1.013	0.999	1.030	1.009
38 x 236	1.000	0.987	1.013	1.009
39 x 242	1.004	1.021	0.995	0.991
40 x 248	1.007	1.006	1.024	1.020
41 x 254	0.999	0.988	1.000	0.994

Tabla A.8: EPP y EMP para diferentes valores de σ en `OSQP`.

	$\sigma = 10^{-10}$		$\sigma = 10^{-8}$		$\sigma = 10^{-4}$		$\sigma = 10^{-2}$	
Dimensión NxM	**EPP**	**EMP**	**EPP**	**EMP**	**EPP**	**EMP**	**EPP**	**EMP**
1 x 14	0.000e+00	0.000e+00	0.000e+00	0.000e+00	1.660e-06	1.659e-04	0.002	0.095
2 x 20	0.000e+00	0.000e+00	0.000e+00	0.000e+00	3.740e-06	2.552e-05	0.001	0.009
3 x 26	1.000e-07	3.370e-06	1.000e-07	3.370e-06	3.070e-06	1.685e-05	0.000	0.001
4 x 32	5.000e-08	3.520e-06	5.000e-08	3.520e-06	4.570e-06	2.162e-04	0.000	0.022
5 x 38	2.400e-07	1.602e-05	2.300e-07	1.602e-05	2.265e-05	0.002	0.001	0.089
6 x 44	1.900e-07	1.092e-05	1.900e-07	1.092e-05	1.810e-05	0.001	0.005	0.396
7 x 50	3.400e-07	2.072e-05	6.300e-07	4.989e-05	2.189e-05	0.001	0.002	0.108
8 x 56	8.000e-08	9.700e-07	8.000e-08	9.700e-07	7.040e-06	7.147e-05	0.001	0.024
9 x 62	9.000e-08	9.200e-07	9.000e-08	9.200e-07	8.230e-06	1.062e-04	0.001	0.050
10 x 68	1.000e-07	3.130e-06	1.000e-07	3.130e-06	9.610e-06	3.149e-04	0.002	0.076
11 x 74	5.000e-08	8.000e-07	5.000e-08	8.000e-07	4.930e-06	8.280e-05	0.001	0.034
12 x 80	8.700e-07	5.087e-05	8.700e-07	5.128e-05	8.652e-05	0.005	0.002	0.098
13 x 86	4.000e-08	6.800e-07	4.000e-08	6.800e-07	4.430e-06	7.145e-05	0.001	0.017
14 x 92	3.300e-07	2.282e-05	3.500e-07	2.519e-05	3.860e-05	0.003	0.005	0.415
15 x 98	5.000e-08	1.830e-06	5.000e-08	1.740e-06	5.280e-06	2.127e-04	0.001	0.021
16 x 104	5.000e-08	5.400e-07	5.000e-08	5.400e-07	4.440e-06	5.382e-05	0.001	0.025
17 x 110	1.100e-07	5.330e-06	1.100e-07	5.390e-06	1.088e-05	5.385e-04	0.001	0.012
18 x 116	5.000e-08	2.480e-06	5.000e-08	2.430e-06	4.400e-06	2.423e-04	0.000	0.024
19 x 122	1.140e-06	7.908e-05	1.220e-06	7.889e-05	1.181e-04	0.008	0.001	0.042
20 x 128	1.400e-07	1.064e-05	1.400e-07	1.047e-05	8.010e-06	4.571e-04	0.001	0.020
21 x 134	3.800e-07	1.966e-05	4.900e-07	2.990e-05	3.740e-05	0.002	0.001	0.027
22 x 140	5.000e-08	2.420e-06	5.000e-08	2.340e-06	5.000e-06	2.159e-04	0.000	0.009
23 x 146	3.900e-07	3.510e-05	3.600e-07	3.213e-05	3.447e-05	0.003	0.009	0.818
24 x 152	1.010e-06	9.592e-05	6.800e-07	6.364e-05	1.396e-04	0.014	0.009	0.880
25 x 158	6.000e-08	3.340e-06	6.000e-08	3.820e-06	6.380e-06	4.024e-04	0.001	0.016
26 x 164	2.000e-07	1.183e-05	2.400e-07	1.720e-05	8.710e-06	3.638e-04	0.001	0.036
27 x 170	5.000e-08	2.000e-06	5.000e-08	1.950e-06	5.020e-06	1.989e-04	0.000	0.009
28 x 176	4.000e-08	1.340e-06	4.000e-08	9.800e-07	4.150e-06	1.102e-04	0.001	0.021
29 x 182	9.300e-07	9.100e-05	9.200e-07	9.000e-05	4.094e-05	0.004	0.004	0.384
30 x 188	1.300e-07	9.920e-06	1.300e-07	9.020e-06	8.910e-06	5.453e-04	0.001	0.039
31 x 194	6.100e-07	5.919e-05	6.100e-07	5.878e-05	0.004	0.408	0.004	0.423
32 x 200	6.000e-08	3.030e-06	5.000e-08	2.690e-06	5.910e-06	3.056e-04	0.003	0.278
33 x 206	3.200e-07	2.051e-05	2.400e-07	2.026e-05	2.691e-05	0.002	0.001	0.020
34 x 212	5.000e-08	2.740e-06	6.000e-08	3.730e-06	5.670e-06	2.999e-04	0.001	0.036
35 x 218	2.000e-08	2.400e-07	2.000e-08	2.200e-07	1.900e-06	2.331e-05	0.001	0.032
36 x 224	2.000e-08	3.000e-07	2.000e-08	3.000e-07	1.940e-06	3.005e-05	0.000	0.003
37 x 230	1.300e-07	1.106e-05	5.000e-08	2.390e-06	1.509e-05	0.001	0.001	0.029
38 x 236	2.500e-07	2.500e-07	2.400e-07	2.400e-07	3.400e-05	3.400e-05	0.001	0.031
39 x 242	9.117e-05	9.117e-05	5.789e-05	5.789e-05	0.001	0.001	0.001	0.084
40 x 248	1.514e-05	1.514e-05	1.448e-05	1.448e-05	0.001	0.001	0.011	1.039
41 x 254	5.554e-05	5.554e-05	4.553e-05	4.553e-05	0.001	0.001	0.001	0.082

A.5. Parámetro ρ

Tabla A.9: Speedups respecto a `OSQP` base para distintos valores del parámetro ρ.

Dimensión NxM	ρ **= 0.01**	ρ **= 0.1**	ρ **= 1**	ρ **= 10**
1 x 14	1.027	1.024	0.832	0.863
2 x 20	1.085	1.067	0.843	0.829
3 x 26	1.040	1.040	0.812	0.731
4 x 32	0.625	0.972	0.883	0.696
5 x 38	0.306	0.789	0.770	0.432
6 x 44	0.239	0.745	0.925	0.395
7 x 50	0.240	0.694	1.020	0.434
8 x 56	0.198	0.604	0.931	0.357
9 x 62	0.196	0.581	1.002	0.390
10 x 68	0.186	0.703	0.857	0.362
11 x 74	0.218	0.700	1.089	0.347
12 x 80	0.217	0.592	1.061	0.407
13 x 86	0.196	0.692	1.081	0.407
14 x 92	0.219	0.613	1.254	0.465
15 x 98	0.201	0.721	1.284	0.365
16 x 104	0.204	0.720	1.530	0.480
17 x 110	0.237	0.778	1.510	0.481
18 x 116	0.218	0.839	1.091	0.429
19 x 122	0.227	0.696	1.272	0.469
20 x 128	0.195	0.822	1.483	0.416
21 x 134	0.247	0.762	1.483	0.561
22 x 140	0.231	0.793	1.465	0.473
23 x 146	0.235	0.793	1.397	0.468
24 x 152	0.244	0.792	1.543	0.533
25 x 158	0.216	0.762	1.548	0.456
26 x 164	0.292	0.869	1.641	0.599
27 x 170	0.309	0.747	1.877	0.551
28 x 176	0.285	0.807	1.557	0.567
29 x 182	0.332	0.896	1.586	0.546
30 x 188	0.251	0.769	1.498	0.533
31 x 194	0.331	0.964	1.967	0.711
32 x 200	0.362	0.949	1.885	0.734
33 x 206	0.282	0.761	1.715	0.675
34 x 212	0.298	0.724	2.032	0.674
35 x 218	0.305	0.790	2.049	0.646
36 x 224	0.301	0.768	1.936	0.654
37 x 230	0.344	0.797	2.139	0.818
38 x 236	0.281	0.833	2.029	0.662
39 x 242	0.321	0.820	1.703	0.678
40 x 248	0.319	0.858	1.588	0.566
41 x 254	0.349	0.849	1.983	0.758

Tabla A.10: EPP y EMP respecto a OSQP base para distintos valores del parámetro ρ.

	$\rho = 0.01$		$\rho = 0.1$		$\rho = 1$		$\rho = 10$	
Dimensión NxM	**EPP**	**EMP**	**EPP**	**EMP**	**EPP**	**EMP**	**EPP**	**EMP**
1 x 14	0.011	0.079	0.000	0.000	0.012	0.095	0.012	0.095
2 x 20	0.009	0.080	2.000e-07	2.000e-05	0.016	0.156	0.016	0.156
3 x 26	0.012	0.110	4.000e-06	4.000e-04	0.016	0.080	0.018	0.114
4 x 32	0.078	3.503	0.003	0.257	0.032	0.302	0.030	0.257
5 x 38	0.946	48.250	0.024	0.614	0.035	0.258	0.039	0.317
6 x 44	1.215	65.096	0.053	2.409	0.047	0.609	0.053	0.792
7 x 50	1.847	58.164	0.117	8.611	0.060	0.611	0.066	0.552
8 x 56	0.982	30.987	0.030	0.406	0.060	0.342	0.062	0.476
9 x 62	2.026	64.403	0.211	14.682	0.082	0.563	0.093	0.936
10 x 68	0.965	24.250	0.027	0.824	0.075	0.376	0.081	0.467
11 x 74	0.853	19.826	0.009	0.160	0.076	0.520	0.080	0.699
12 x 80	2.509	93.719	0.801	66.547	0.083	0.353	0.100	0.436
13 x 86	1.095	29.720	0.258	19.687	0.109	0.537	0.100	0.391
14 x 92	4.080	293.320	2.015	198.690	0.196	10.039	0.119	2.426
15 x 98	1.119	27.652	0.024	0.955	0.099	0.592	0.113	0.833
16 x 104	1.235	19.056	0.101	4.953	0.108	0.474	0.131	0.572
17 x 110	1.200	18.127	0.115	7.281	0.106	0.377	0.120	0.453
18 x 116	1.143	42.803	0.220	19.121	0.138	2.075	0.152	2.410
19 x 122	1.640	26.997	0.146	7.101	0.148	2.816	0.133	0.531
20 x 128	1.102	44.467	0.119	8.554	0.123	0.381	0.123	0.520
21 x 134	1.345	25.065	0.213	6.859	0.118	0.372	0.136	0.640
22 x 140	1.520	68.192	0.166	8.634	0.130	0.454	0.141	0.454
23 x 146	1.400	29.250	0.311	18.815	0.134	1.167	0.164	2.420
24 x 152	1.564	40.878	0.172	11.546	0.195	6.862	0.161	2.277
25 x 158	1.215	49.597	0.125	8.568	0.139	0.414	0.147	0.430
26 x 164	1.352	16.232	0.131	5.060	0.144	1.194	0.140	0.417
27 x 170	1.365	16.257	0.165	7.876	0.137	0.518	0.147	0.477
28 x 176	1.349	16.066	0.137	6.129	0.162	2.813	0.206	5.122
29 x 182	1.856	39.294	0.113	7.266	0.165	1.610	0.206	4.201
30 x 188	1.371	27.814	0.239	9.804	0.173	1.719	0.189	2.341
31 x 194	1.351	20.699	0.074	3.210	0.165	1.504	0.191	4.515
32 x 200	3.089	64.100	0.435	17.765	0.139	1.035	0.174	1.325
33 x 206	2.004	36.221	0.288	12.702	0.177	2.475	0.252	7.163
34 x 212	2.343	48.272	0.146	4.495	0.157	1.111	0.181	1.390
35 x 218	1.946	29.165	0.105	3.935	0.145	0.969	0.171	1.335
36 x 224	1.748	28.068	0.114	6.088	0.167	2.205	0.199	2.341
37 x 230	2.007	18.617	0.127	4.075	0.172	1.998	0.203	2.411
38 x 236	1.682	39.470	0.029	1.194	0.173	1.301	0.236	6.427
39 x 242	1.797	25.980	0.251	11.534	0.248	3.640	0.273	4.399
40 x 248	1.251	13.792	0.188	5.128	0.177	1.358	0.188	1.721
41 x 254	1.923	16.763	0.079	2.753	0.221	2.957	0.232	3.213

A.6. Parámetro check_termination

Tabla A.11: Speedups respecto a `OSQP` base para distintos valores del parámetro `check_termination` (CT).

Dimensión NxM	**CT = 5**	**CT = 10**	**CT = 50**	**CT = 100**
1 x 14	0.976	0.802	0.928	0.867
2 x 20	0.976	0.809	0.928	0.840
3 x 26	0.981	0.778	0.935	0.811
4 x 32	0.971	0.760	0.933	0.817
5 x 38	0.890	0.714	0.933	0.833
6 x 44	0.911	0.757	0.915	0.814
7 x 50	1.005	0.877	1.058	0.942
8 x 56	0.887	0.778	0.940	0.854
9 x 62	0.857	0.706	0.924	0.809
10 x 68	0.906	0.765	0.940	0.852
11 x 74	0.923	0.789	0.966	0.854
12 x 80	0.937	0.908	1.017	0.920
13 x 86	0.893	0.864	0.997	0.906
14 x 92	0.848	0.846	0.920	0.884
15 x 98	0.835	0.864	0.953	0.880
16 x 104	0.872	0.859	0.950	0.908
17 x 110	0.828	0.848	0.953	0.902
18 x 116	0.837	0.855	0.956	0.898
19 x 122	0.854	0.883	0.956	0.906
20 x 128	0.905	0.888	0.947	0.902
21 x 134	0.818	0.830	0.936	0.923
22 x 140	0.824	0.893	0.917	0.870
23 x 146	0.868	0.904	0.980	0.938
24 x 152	0.886	0.893	0.957	0.903
25 x 158	0.816	0.856	0.947	0.876
26 x 164	0.899	0.936	1.039	1.006
27 x 170	0.794	0.963	0.934	0.904
28 x 176	0.872	0.926	1.005	0.960
29 x 182	0.826	0.853	0.952	0.893
30 x 188	0.811	0.901	0.955	0.924
31 x 194	0.945	1.026	1.112	1.080
32 x 200	0.989	1.084	1.127	1.114
33 x 206	0.842	0.916	0.994	0.941
34 x 212	0.825	0.889	0.961	0.940
35 x 218	0.850	0.933	0.988	0.960
36 x 224	0.853	0.922	1.007	0.975
37 x 230	0.861	0.920	1.016	0.997
38 x 236	0.832	0.904	0.985	0.977
39 x 242	0.846	0.932	0.994	0.989
40 x 248	0.875	0.920	1.008	0.965
41 x 254	0.824	0.908	1.016	0.987

Tabla A.12: EPP y EMP para distintos valores de `check_termination` (CT).

	CT = 5		**CT = 10**		**CT = 50**		**CT = 100**	
Dimensión NxM	**EPP**	**EMP**	**EPP**	**EMP**	**EPP**	**EMP**	**EPP**	**EMP**
1 x 14	1.487e-03	0.023	0.013	0.099	0.012	0.095	0.012	0.095
2 x 20	1.386e-04	0.010	0.017	0.166	0.016	0.156	0.016	0.156
3 x 26	1.698e-04	0.014	0.018	0.085	0.017	0.080	0.017	0.080
4 x 32	0.043	3.594	0.022	0.251	0.018	0.116	0.025	0.241
5 x 38	0.021	0.199	0.026	0.157	0.015	0.240	0.018	0.240
6 x 44	0.029	0.169	0.030	0.169	0.012	0.139	0.022	0.229
7 x 50	0.035	0.403	0.035	0.361	0.019	0.287	0.041	0.389
8 x 56	0.029	0.212	0.029	0.212	0.017	0.149	0.031	0.282
9 x 62	0.034	0.222	0.035	0.222	0.029	0.156	0.049	0.212
10 x 68	0.029	0.164	0.031	0.265	0.023	0.304	0.045	0.312
11 x 74	0.027	0.122	0.027	0.122	0.023	0.195	0.045	0.288
12 x 80	0.026	0.126	0.022	0.126	0.023	0.136	0.050	0.244
13 x 86	0.026	0.156	0.022	0.110	0.017	0.127	0.038	0.185
14 x 92	0.035	0.281	0.028	0.173	0.015	0.097	0.034	0.410
15 x 98	0.031	0.194	0.028	0.158	0.023	0.146	0.044	0.213
16 x 104	0.027	0.095	0.023	0.095	0.021	0.141	0.047	0.216
17 x 110	0.030	0.179	0.027	0.179	0.017	0.135	0.039	0.208
18 x 116	0.031	0.128	0.027	0.128	0.024	0.196	0.047	0.272
19 x 122	0.031	0.170	0.022	0.170	0.016	0.131	0.042	0.274
20 x 128	0.032	0.172	0.028	0.124	0.030	0.205	0.053	0.294
21 x 134	0.027	0.147	0.021	0.098	0.020	0.113	0.048	0.186
22 x 140	0.027	0.139	0.024	0.121	0.023	0.187	0.051	0.187
23 x 146	0.030	0.129	0.026	0.100	0.026	0.118	0.048	0.189
24 x 152	0.026	0.123	0.022	0.108	0.022	0.190	0.048	0.227
25 x 158	0.025	0.105	0.022	0.098	0.022	0.187	0.055	0.198
26 x 164	0.026	0.270	0.022	0.286	0.021	0.182	0.052	0.183
27 x 170	0.029	0.126	0.022	0.093	0.024	0.097	0.042	0.175
28 x 176	0.026	0.131	0.022	0.115	0.022	0.131	0.050	0.193
29 x 182	0.030	0.146	0.023	0.136	0.017	0.098	0.043	0.182
30 x 188	0.025	0.127	0.020	0.127	0.014	0.128	0.045	0.177
31 x 194	0.028	0.134	0.022	0.134	0.015	0.104	0.042	0.163
32 x 200	0.028	0.104	0.023	0.104	0.022	0.158	0.045	0.246
33 x 206	0.022	0.099	0.016	0.081	0.014	0.094	0.036	0.135
34 x 212	0.022	0.117	0.019	0.117	0.015	0.114	0.036	0.182
35 x 218	0.025	0.139	0.019	0.085	0.019	0.071	0.043	0.133
36 x 224	0.024	0.132	0.020	0.132	0.015	0.106	0.037	0.181
37 x 230	0.019	0.150	0.015	0.094	0.018	0.112	0.036	0.133
38 x 236	0.025	0.094	0.021	0.080	0.017	0.098	0.038	0.171
39 x 242	0.022	0.115	0.018	0.115	0.016	0.119	0.043	0.180
40 x 248	0.027	0.112	0.022	0.112	0.018	0.133	0.040	0.215
41 x 254	0.021	0.110	0.017	0.110	0.021	0.143	0.044	0.326

A.7. Caracterización en entorno embebido

Tabla A.13: Speedups respecto a `OSQP` en computador portátil al ejecutar problemas base en Raspberry Pi 4 y Raspberry Pi 5.

Dimensión Problema	**S Raspberry Pi 4**	**S Raspberry Pi 5**
1 x 14	0.540	1.616
2 x 20	0.527	1.572
3 x 26	0.462	1.343
4 x 32	0.448	1.304
5 x 38	0.340	0.990
6 x 44	0.286	0.817
7 x 50	0.314	0.908
8 x 56	0.262	0.780
9 x 62	0.236	0.756
10 x 68	0.253	0.745
11 x 74	0.234	0.737
12 x 80	0.250	0.755
13 x 86	0.242	0.710
14 x 92	0.213	0.722
15 x 98	0.200	0.669
16 x 104	0.214	0.695
17 x 110	0.212	0.734
18 x 116	0.220	0.677
19 x 122	0.235	0.722
20 x 128	0.237	0.710
21 x 134	0.261	0.789
22 x 140	0.264	0.777
23 x 146	0.214	0.720
24 x 152	0.258	0.750
25 x 158	0.270	0.764
26 x 164	0.278	0.797
27 x 170	0.289	0.824
28 x 176	0.276	0.778
29 x 182	0.302	0.836
30 x 188	0.284	0.778
31 x 194	0.298	1.019
32 x 200	0.322	0.900
33 x 206	0.272	0.888
34 x 212	0.295	0.871
35 x 218	0.295	0.895
36 x 224	0.290	0.899
37 x 230	0.332	1.023
38 x 236	0.293	0.896
39 x 242	0.300	0.959
40 x 248	0.328	1.022
41 x 254	0.294	0.982

Tabla A.14: EPP y EMP respecto a OSQP en computador portátil, para problemas base ejecutados en Raspberry Pi.

	Raspberry Pi vs OSQP (PC)	
Dimensión NxM	**EPP**	**EMP**
1 x 14	0.003	0.080
2 x 20	1.429e-05	0.001
3 x 26	1.425e-04	0.014
4 x 32	0.357	15.298
5 x 38	0.021	0.394
6 x 44	0.069	2.516
7 x 50	0.112	2.712
8 x 56	0.083	1.177
9 x 62	0.202	4.457
10 x 68	0.098	1.985
11 x 74	0.120	1.850
12 x 80	0.218	4.601
13 x 86	0.197	3.353
14 x 92	0.417	27.330
15 x 98	0.172	2.516
16 x 104	0.267	4.314
17 x 110	0.208	2.704
18 x 116	0.269	5.411
19 x 122	0.304	3.407
20 x 128	0.223	4.968
21 x 134	0.418	6.326
22 x 140	0.334	4.060
23 x 146	0.402	8.035
24 x 152	0.344	6.741
25 x 158	0.369	5.454
26 x 164	0.400	4.197
27 x 170	0.416	5.482
28 x 176	0.445	4.570
29 x 182	0.509	7.885
30 x 188	0.465	6.711
31 x 194	0.497	5.233
32 x 200	0.721	12.372
33 x 206	0.582	5.482
34 x 212	0.674	7.845
35 x 218	0.617	4.029
36 x 224	0.581	5.434
37 x 230	0.792	3.880
38 x 236	0.566	3.453
39 x 242	0.731	6.761
40 x 248	0.623	5.932
41 x 254	0.751	4.140

Bibliografía

[1] 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, 2020. [Online]. Available: https://doi.org/10.1007/s12532-020-00179-2

[2] S. González, "Implementación de algoritmo de control predictivo para motor de corriente continua utilizando sistemas embebidos," Memoria de titulo, Universidad Técnica Federico Santa María, 2022.

[3] J. Rawlings, D. Mayne, and M. Diehl, *Model Predictive Control: Theory, Computation, and Design*. Nob Hill Publishing, 2017.

[4] MathWorks, *quadprog - Quadratic Programming Solver*, 2020. [Online]. Available: https://www.mathworks.com/help/optim/ug/quadprog.html

[5] A. P. H. G. B. M. D. Hans Joachim Ferreau, Christian Kirches, "qpoases: A parametric active set algorithm for quadratic programming," *Computational Optimization and Applications*, 2014.

[6] "Gurobi optimizer: The fastest solver for mathematical programming," https://www.gurobi.com, 2025, accessed: 2025-03-17.

[7] "Mosek: Optimization software for solving linear, quadratic, and conic programming problems," https://www.mosek.com, 2025, accessed: 2025-03-17.

[8] J. Kim and K. S. Lee, "Efficient quadratic programming algorithms for model predictive control," in *2012 12th International Conference on Control, Automation and Systems*, 2012, pp. 553–555.

[9] D. Figueroa, "Modelado y simulación de recursos energéticos distribuidos (der) en microrredes," Master's thesis, Universidad Técnica Federico Santa María, Valparaíso, Chile, 2024.

[10] I. Acosta, "Reinforcement learning-based primary control for der units integration in ac microgrids," Master's thesis, Universidad Técnica Federico Santa María, Valparaíso, Chile, 2024.

[11] M. Simon, "TiempoMPC," https://github.com/manuelsimonv/TiempoMPC, 2025, gitHub repository. Accessed: 2025-10-19.

[12] A. Cortés, "Implementación de un controlador predictivo basado en fpga para control en tiempo real," Master's thesis, Universidad Técnica Federico Santa María, Chile, 2023.

[13] J. Nocedal and S. J. Wright, *Numerical Optimization*, 2nd ed., ser. Springer Series in Operations Research and Financial Engineering. New York: Springer, 2006.

[14] D. Mayne and H. Michalska, "Receding horizon control of nonlinear systems," *IEEE Transactions on Automatic Control*, vol. 35, no. 7, pp. 814–824, 1990.

[15] K. Zhang, D. Zhou, Z. Yang, X. Li, Y. Zhao, and W. Kong, "A dynamic weapon target assignment based on receding horizon strategy by heuristic algorithm," in *Journal of Physics: Conference Series*, vol. 1651. IOP Publishing, Nov. 2020, p. 012062.

[16] F. Borrelli, A. Bemporad, and M. Morari, *Predictive Control for Linear and Hybrid Systems*. Cambridge University Press, 2017.

[17] S. J. Wright, *Primal-Dual Interior-Point Methods*. SIAM, 1997.

[18] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, *Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers*, 2011.

[19] OSQP Developer Team. (2025) Solver settings — osqp documentation. OSQP. [Online]. Available: https://osqp.org/docs/interfaces/solver_settings.html

[20] G. Cimini, A. Bemporad, and D. Bernardini, "Odys qp solver," ODYS S.r.l., Tech. Rep., Sep. 2017.