



UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE CIENCIAS

REDES NEURONALES ARTIFICIALES PARA RESOLVER LA  
ECUACIÓN DE SCHRÖDINGER DEPENDIENTE DEL TIEMPO

---

T E S I S

Que para obtener el título de:

Física

Presenta:

ITZEL JESSICA MARTÍNEZ MARCELO

Tutor:

Dr. Huziel Enoc Saucedo Felix

2023

Itzel Jessica Martínez Marcelo : *Redes Neuronales Artificiales para resolver la Ecuación de Schrödinger Dependiente del Tiempo*, UNAM Facultad de Ciencias, © 2023

TUTOR:

Dr. Huziel Enoc Saucedo Felix

UBICACIÓN:

Ciudad de México, México



## RESUMEN

La ecuación de Schrödinger describe la evolución temporal de sistemas cuánticos, cuando estos sistemas involucran interacciones físicas y químicas que dependen del tiempo la resolución de la ecuación puede volverse más complicada y compleja para los métodos analíticos o costosa en tiempo computacional para los métodos numéricos. Las redes neuronales artificiales ([ANNs](#)) son un método de aprendizaje automático que ha ganado popularidad en los últimos años debido a los avances en computación y a su versatilidad en aplicaciones a diversas áreas, incluidas las ciencias. En este trabajo se presenta una alternativa a los métodos convencionales para resolver la ecuación de Schrödinger dependiente del tiempo y encontrar la evolución temporal en sistemas de transferencia de protones, mediante el uso de una [ANN](#) de tipo long short-term memory, que propaga un paquete de onda inicial bajo un potencial dependiente del tiempo en pasos de tiempo cortos durante un periodo largo de tiempo, considerando la escala de tiempo característico que tienen este tipo de procesos químicos y biológicos. Se encontró que la red entrenada puede propagar paquetes de onda utilizando un tiempo de ejecución menor al de otros métodos numéricos, proponiendo así una alternativa viable para la resolución de estos sistemas.



## ABSTRACT

The Schrödinger equation describes the time evolution of quantum systems. When these systems involve time-dependent physical and chemical interactions, solving the equation can become more complicated and complex for analytical methods or computationally expensive for numerical methods. Artificial neural networks ([ANNs](#)) are a machine learning method that has gained popularity recently due to technological advances and its versatility in applications to various areas, including the sciences. In this work, we present an alternative to conventional methods for solving the time-dependent Schrödinger equation and finding the time evolution in proton transfer systems by using a long short-term memory type [ANN](#), which propagates an initial wave packet under a time-dependent potential in short time steps over a long period, considering the characteristic time scale of this chemical and biological processes. We found that the trained network can propagate wave packets using a shorter execution time than other numerical methods, thus proposing a viable alternative for the resolution of these systems.

*A mis padres y hermanas*



## AGRADECIMIENTOS

A mi asesor, el Dr. Huziel E. Sauceda, por apoyarme y ofrecerme siempre su mentoría durante el desarrollo de este proyecto, sus consejos e ideas fueron esenciales para la culminación de este trabajo.

A mis sinodales, por tomarse el tiempo de leer y evaluar mi trabajo.

A mis buenos profesores y compañeros, que fomentaron un entorno humanístico tan importante como el académico. En particular al Dr. José Rubén Alfaro, José Serna, Valeria, Sergio y Daniel, quienes fueron un equipo importante por brindarme ratos de mucha diversión, y apoyo en momentos difíciles.

A Edson por ser mi gran camarada de clases.

A mis mejores amigos Diana y Carlos, por siempre estar, por sus consejos y compañía.

Mis años como estudiante universitaria representaron un verdadero reto, y estaré eternamente agradecida con mi familia por la comprensión y el apoyo que siempre me brindaron. A mis padres, Irma y Antonio, sin duda todos mis pequeños y grandes logros son gracias a su guía, esfuerzo y al infinito amor con el que me criaron. A mis hermanas, Nelly y Aylen, porque siempre dieron más de ellas en el hogar para que yo pudiera dedicarme únicamente a estudiar, su compañía y alegría fueron mi refugio en los momentos más difíciles y me dieron la fuerza para no rendirme. A mis abuelitos, tíos, tíos y primas, porque la universidad llegó a consumirme mucho tiempo que no pude dedicarle a ellos, ni estar en muchos momentos importantes, y aún así, el apoyo, la calidez y amor cuando nos reunimos están intactos.

A Kotomi y Carmina, mis lomitos, por existir.

Por último, quiero agradecer al compañero de mi clase de mecánica que un día me pasó apuntes, Aldo, porque desde el día que lo conocí, su pasión por la Física me inspiró y motivó a seguir este camino, y porque cuando el camino se volvía oscuro siempre se convertía en luz; por siempre acompañarme y darme seguridad, por convertirse en el mejor amigo, el mejor novio y el mejor esposo que la vida me pudo dar.



## ÍNDICE GENERAL

### I PRÓLOGO

1	INTRODUCCIÓN	2
1.1	Organización . . . . .	3

### II ECUACIÓN DE SCHRÖDINGER DEPENDIENTE DEL TIEMPO

2	CONCEPTOS FUNDAMENTALES DE MECÁNICA CUÁNTICA	6
2.1	Estados y operadores . . . . .	6
2.2	Representación matricial de un operador . . . . .	8
2.3	Operador de propagación: evolución temporal de un sistema . . . . .	8
2.4	Función de onda en el espacio de posiciones . . . . .	10
2.4.1	Densidad de probabilidad . . . . .	11
3	REPRESENTACIÓN DE VARIABLE DISCRETA	12
3.1	Proyección espectral y colocación . . . . .	12
3.1.1	Representación de la función de onda bajo el operador de proyección . . . . .	15
3.1.2	Definición del proyector de colocación . . . . .	15
3.1.3	Colocación ortogonal . . . . .	16
3.2	Base pseudo-espectral . . . . .	17
3.2.1	Completes y ortogonalidad de las funciones base pseudo-espectrales . . . . .	18
3.2.2	El proyector de colocación en la base pseudo-espectral . . . . .	19
3.3	Algoritmo DVR aplicado a un proceso físico-químico . . . . .	19
3.3.1	Sistema de transferencia de protones . . . . .	20
3.3.2	Elección de una base ortonormal y malla . . . . .	22
3.3.3	Representación del operador espacial $\hat{r}$ en la base espectral	24
3.3.4	Diagonalización mediante una transformación unitaria . . . . .	24
3.3.5	Representación del Hamiltoniano en la base pseudo-espectral	25
3.3.6	Propagación de un paquete de onda . . . . .	26

### III REDES NEURONALES ARTIFICIALES (ANNS)

4	APRENDIZAJE AUTOMÁTICO	31
4.1	Aprendizaje supervisado . . . . .	32
4.2	Aprendizaje no supervisado . . . . .	33

4.3 Aprendizaje profundo . . . . .	33
<b>5 REDES NEURONALES ARTIFICIALES</b>	<b>34</b>
5.1 Introducción . . . . .	34
5.2 Arquitectura básica . . . . .	35
5.2.1 El perceptrón . . . . .	35
5.2.2 Función de activación . . . . .	36
5.2.3 Función de pérdida . . . . .	37
5.2.4 Modelos multicapa . . . . .	37
5.3 Entrenamiento de una red neuronal . . . . .	38
 <b>IV IMPLEMENTACIÓN DE UN MODELO DE LSTM COMO PROPAGADOR EN DINÁMICA CUÁNTICA</b>	
<b>6 REDES LONG SHORT-TERM MEMORY PARA RESOLVER LA ECUACIÓN DE SCHRÖDINGER</b>	<b>42</b>
6.1 Introducción . . . . .	42
6.2 Redes neuronales recurrentes . . . . .	43
6.3 Redes long short-term memory . . . . .	44
6.3.1 Arquitectura . . . . .	45
6.4 Propagación temporal de funciones de onda con LSTM . . . . .	46
6.4.1 Obtención de datos . . . . .	46
6.4.2 Procesamiento de datos: visualización y forma . . . . .	50
6.4.3 Modelo LSTM: arquitectura y entrenamiento . . . . .	51
6.4.4 Resultados y análisis: precisión y predicciones . . . . .	53
6.4.5 Aumento de resolución con dos LSTM . . . . .	61
6.4.6 Conclusión . . . . .	65
 <b>V APÉNDICE</b>	
<b>A APÉNDICE</b>	<b>67</b>
A.1 Prueba de la identidad . . . . .	67
A.2 Variables de parámetros para el potencial . . . . .	67
A.3 Ejemplos de potenciales utilizados . . . . .	69
A.4 Unidades atómicas (a.u.) . . . . .	71
<b>BIBLIOGRAFÍA</b>	<b>72</b>



## ÍNDICE DE FIGURAS

Figura 3.1	Interpretación geométrica de una proyección ortogonal y una no ortogonal del un vector en el espacio $\mathbb{R}^3$ al espacio $\mathbb{R}^2$ . . . . .	16
Figura 3.2	Descripción del modelo de transferencia de protones $H^+$ . . . . .	20
Figura 3.3	Potencial $V(r, t)$ a diferentes tiempos. . . . .	21
Figura 3.4	Estado base (GS), primer estado excitado (ES) y potenciales armónicos $U_1(r, (R(t)))$ y $U_2(r, R(t))$ al tiempo $t = 0$ . . . . .	22
Figura 3.5	Funciones ortogonales $\phi_n(r)$ en la malla $\{r_i\}_{i=1}^N$ para $n = 1, \dots, 5$ . . . . .	23
Figura 3.6	Funciones ortogonales $\theta_\alpha(r)$ en la malla $\{r_i\}_{i=1}^N$ para $\alpha = 1, 10, 22, 32, 42$ . . . . .	26
Figura 3.7	Paquete inicial de onda al tiempo $t = 0$ . Parte real y compleja. . . . .	27
Figura 3.8	Propagación del paquete de onda $\psi(r, t)$ parte real. . . . .	28
Figura 3.9	Propagación del paquete de onda $\psi(r, t)$ parte imaginaria. . . . .	28
Figura 3.10	Evolución temporal de la densidad $ \psi(r, t) ^2$ . . . . .	29
Figura 4.1	Diagrama de Venn que muestra los conceptos de inteligencia artificial, aprendizaje automático y aprendizaje profundo. . . . .	31
Figura 5.1	Neuronas biológicas transmitiendo información [8] . . . . .	34
Figura 5.2	Diagrama de un modelo simple de perceptrón . . . . .	35
Figura 5.3	Ejemplos de funciones de activación . . . . .	36
Figura 5.4	Diagrama de un modelo de ANN multicapa con dos capas ocultas. . . . .	38
Figura 5.5	Algoritmo de descenso del gradiente para una función de pérdida $L(w)$ . . . . .	39
Figura 5.6	Gráfica computacional con dos posibles caminos [1] . . . . .	40
Figura 6.1	Diagrama de una RNN y su representación en capas de tiempo. . . . .	44
Figura 6.2	Diagrama de una LSTM representada en capas de tiempo. . . . .	44
Figura 6.3	Diagrama de un bloque de memoria de una LSTM. . . . .	45
Figura 6.4	Diagrama de una trayectoria generada. Para cada tiempo $t_i$ la onda $\psi(r, t_i)$ contiene el valor de la onda en los $N = 32$ puntos en el grid del espacio de posiciones en $r$ . . . . .	47
Figura 6.5	Parte real y compleja de un paquete de onda inicial $\psi(r, t = 0)$ . . . . .	47

Figura 6.6	Potencial inicial al tiempo $t = 0$ . . . . .	48
Figura 6.7	Visualización gráfica: Entrada $X$ y etiqueta $y$ a un tiempo $t$ . . . . .	51
Figura 6.8	Diagrama del modelo LSTM: se muestran 3 de las 200 capas de tiempo. . . . .	52
Figura 6.9	Valores promedio de $ S $ y $\theta$ en el conjunto de validación por cada época. . . . .	54
Figura 6.10	Ejemplo 1 de predicción a un paso de tiempo $\Delta t = 1\text{fs}$ . Las funciones de onda $\psi$ están escaladas para poder visualizar el potencial en la misma gráfica. . . . .	54
Figura 6.11	Ejemplo 2 de predicción a un paso de tiempo $\Delta t = 1\text{fs}$ . Las funciones de onda $\psi$ están escaladas para poder visualizar el potencial en la misma gráfica. . . . .	55
Figura 6.12	Ejemplo 3 de predicción a un paso de tiempo $\Delta t = 1\text{fs}$ . Las funciones de onda $\psi$ están escaladas para poder visualizar el potencial en la misma gráfica. . . . .	55
Figura 6.13	Ejemplo 4 de predicción a un paso de tiempo $\Delta t = 1\text{fs}$ . Las funciones de onda $\psi$ están escaladas para poder visualizar el potencial en la misma gráfica. . . . .	56
Figura 6.14	Ejemplo 5 de predicción a un paso de tiempo $\Delta t = 1\text{fs}$ . Las funciones de onda $\psi$ están escaladas para poder visualizar el potencial en la misma gráfica. . . . .	56
Figura 6.15	Ejemplo 1 de predicción para una trayectoria completa de $200\text{fs}$ . Las densidades de probabilidad $ \psi $ están escaladas para poder visualizar el potencial en la misma gráfica. . . . .	58
Figura 6.16	Ejemplo 2 de predicción para una trayectoria completa de $200\text{fs}$ . Las densidades de probabilidad $ \psi $ están escaladas para poder visualizar el potencial en la misma gráfica . . . . .	59
Figura 6.17	Ejemplo 3 de predicción para una trayectoria completa de $200\text{fs}$ . Las densidades de probabilidad $ \psi $ están escaladas para poder visualizar el potencial en la misma gráfica . . . . .	60
Figura 6.18	División de 64 puntos en el espacio de posiciones en dos grupos. . . . .	61
Figura 6.19	Ejemplo 1 de predicción a un paso de tiempo $\Delta t = 1\text{fs}$ , $N = 64$ puntos. Las funciones de onda $\psi$ están escaladas para poder visualizar el potencial en la misma gráfica. . . . .	62
Figura 6.20	Ejemplo 2 de predicción a un paso de tiempo $\Delta t = 1\text{fs}$ , $N = 64$ puntos. Las funciones de onda $\psi$ están escaladas para poder visualizar el potencial en la misma gráfica. . . . .	62

Figura 6.21	Ejemplo 3 de predicción a un paso de tiempo $\Delta t = 1fs$ , $N = 64$ puntos. Las funciones de onda $\psi$ están escaladas para poder visualizar el potencial en la misma gráfica. . . . .	63
Figura 6.22	Ejemplo de predicción para una trayectoria completa de $200fs$ , $N = 64$ puntos. Las densidades de probabilidad $ \psi $ están escaladas para poder visualizar el potencial en la misma gráfica . . . . .	64
Figura A.1	Ejemplos de potenciales utilizados $V(r, t)$ . . . . .	69
Figura A.2	Ejemplos de potenciales utilizados $V(r, t)$ . . . . .	70



## ÍNDICE DE CUADROS

Tabla 6.1	División de datos . . . . .	51
Tabla 6.2	Resumen del modelo . . . . .	53
Tabla 6.3	Valores promedio de $ S $ y $\theta$ en el conjunto de testeo . . . . .	54
Tabla 6.4	Tiempo requerido para propagar un paquete de onda un paso en el tiempo $\Delta t$ . . . . .	57
Tabla A.1	Valores de parámetros del potencial utilizados para generar la gráfica de la Figura 3.3 . . . . .	67
Tabla A.2	Rango de valores de parámetros del potencial utilizados para generar trayectorias. . . . .	68
Tabla A.3	Conversión de unidades atómicas a unidades SI. . . . .	71
Tabla A.4	Conversión de energía (diversas unidades). . . . .	71



## ACRÓNIMOS

TDSE	Ecuación de Schrödinger Dependiente del Tiempo (Time Dependent Schrödinger Equation)
------	--

DVR	Representación de Variable Discreta (Discrete Variable Representation)
ANN	Red Neuronal Artificial (Artificial Neural Network)
RNN	Red Neuronal Recurrente (Recurrent Neural Network)
LSTM	(Long Short-Term Memory)

Parte I

## PRÓLOGO

# 1



## INTRODUCCIÓN

La mecánica cuántica es un área de la física que describe sistemas a escalas atómicas. Su entendimiento fue cuestionado por las personas de la época en sus inicios (1920's), pues sus fundamentos eran ideas revolucionarias y contra intuitivas comparadas con la descripción de la naturaleza que ofrecía la mecánica que hasta entonces se conocía: la mecánica clásica, pues con la información suficiente de un sistema físico, esta ofrecía respuestas concretas al predecir su evolución en el tiempo. La mecánica cuántica por otro lado, plantea sus respuestas en términos de una densidad de probabilidad, en donde, hasta hacer una medición, se puede saber con exactitud la variable física del sistema que se quiere estudiar.

La interpretación de la mecánica cuántica sigue siendo un tema que genera discusión, sin embargo, la validez y fortaleza de la teoría no son puestas en duda, pues es capaz de explicar el comportamiento y las interacciones de las partículas a pequeña escala de manera exitosa y de acuerdo a las pruebas experimentales.

La evolución temporal de un sistema cuántico es de gran interés para diferentes áreas de la ciencia, pues tiene diversas aplicaciones para procesos químicos y biológicos. En tales procesos es importante tener una descripción cuántica del sistema, ya que trata de interacciones a niveles atómicos o moleculares. La ecuación de Schrödinger dependiente del tiempo, ([TDSE](#), por sus siglas en inglés), es la ecuación que describe la evolución temporal de un sistema cuántico; por otro lado, el Hamiltoniano de un sistema, en el formalismo de la mecánica cuántica, es un operador que contiene la información de la energía total del sistema.

El Hamiltoniano caracteriza completamente a un sistema físico, por lo tanto, la solución a la ecuación de Schrödinger depende del Hamiltoniano. Muchos procesos químicos implican una dependencia temporal, lo que conduce a un Hamiltoniano complejo. Para resolver la [TDSE](#) y encontrar una descripción de la evolución del sistema, se pueden emplear diversos métodos analíticos o numéricos; sin embargo, dependiendo de la naturaleza del sistema, los cálculos o su tiempo de cómputo pueden llegar a representar un problema para obtener resultados de manera eficiente.

En los últimos años, las redes neuronales artificiales ([ANNs](#), por sus siglas en inglés)

han sido modelos de aprendizaje automático que han cobrado popularidad debido a los avances en computación, al uso de unidades de procesamiento gráfico (GPU's, por sus siglas en inglés), al mejoramiento en algoritmos y modelos; y al incremento de datos. Actualmente pueden ofrecer resultados que consumen menos tiempo y/o memoria de cómputo en diversos problemas con una precisión exitosa. De la manera más general, las **ANNs** pueden verse como modelos computacionales, cuya forma de aprender<sup>1</sup> se basa en procesar ejemplos de datos referentes al problema o sistema en cuestión.

El objetivo de este trabajo es implementar una **ANN** que funcione como propagador para cierto tipo de sistemas cuánticos dependientes del tiempo. Es decir, una **ANN** que mapee en un periodo largo de tiempo  $\{t_0, t_1, \dots, t_n\}$  una función de onda en un tiempo inicial  $\psi(r, t)$  a un intervalo de tiempo posterior  $\psi(r, t + \Delta t)$  bajo potencias que dependen del tiempo, obteniendo así un modelo alternativo a los métodos analíticos y numéricos para la resolución de la **TDSE** para este tipo de sistemas.

### 1.1 ORGANIZACIÓN

En la segunda parte de la tesis (**Parte ii**) se abordan conceptos básicos de dinámica cuántica y la ecuación de Schrödinger dependiente del tiempo para la descripción de la evolución temporal de sistemas cuánticos. En el **Capítulo 3** se desarrolla un ejemplo de método numérico para la resolución de la **TDSE** aplicada al tipo de sistemas cuánticos de interés para este trabajo.

La tercera parte de la tesis (**Parte iii**) describe una breve introducción sobre aprendizaje automático y aprendizaje supervisado, posteriormente se describen los fundamentos y características principales de las redes neuronales artificiales (**ANNs**).

La última parte de la tesis (**Parte iv**) comienza describiendo las redes neuronales recurrentes, (**RNNs**, por sus siglas en inglés), un tipo de redes neuronales artificiales que es útil para abordar sistemas descritos en secuencias de tiempo.

En la sección 6.4 se implementa una red neuronal recurrente para predecir la evolución de un paquete de onda bajo un potencial dependiente del tiempo característico de los sistemas de transferencia de protones (**§ 3.3.1**). En las secciones **§ 6.4.1** y **§ 6.4.2** se desarrollan los métodos computacionales utilizados para la generación y procesamiento de los datos de entrenamiento, en la sección **§ 6.4.3** se describen los detalles de la red neuronal artificial relacionados con su arquitectura y entrena-

---

<sup>1</sup> Históricamente las **ANNs** fueron inspiradas por las estructuras neuronales biológicas; un concepto como el *entrenamiento* de una **ANN**, puede interpretarse como el proceso análogo de *aprendizaje* que tienen los organismos biológicos.

miento. Finalmente, en la sección § 6.4.4 se muestra la precisión del modelo y sus predicciones, así como sugerencias para implementar en trabajos futuros relacionados.

## Parte II

# ECUACIÓN DE SCHRÖDINGER DEPENDIENTE DEL TIEMPO

# 2



## CONCEPTOS FUNDAMENTALES DE MECÁNICA CUÁNTICA

El formalismo matemático general para trabajar en mecánica cuántica es el álgebra lineal. En las siguientes secciones se desarrollan algunos de los conceptos fundamentales en mecánica cuántica utilizando la notación de *bra-ket*, introducida por primera vez por Paul Dirac.

### 2.1 ESTADOS Y OPERADORES

La información física de un sistema en mecánica cuántica está contenida en su **estado** o **ket**, que en general es un elemento del espacio de Hilbert. Los **espacios de Hilbert** son espacios vectoriales reales o complejos que cuentan con un producto punto y pueden ser de dimensión infinita. Este espacio es conocido como el **espacio ket**.

$$|\psi\rangle$$

La dimensión del espacio depende de las diferentes posibilidades o alternativas que el sistema puede tomar, por ejemplo, si el sistema es un átomo y las medición que se está realizando es de su spin, el átomo, que se representa por su estado, pertenece a un espacio vectorial de dos dimensiones, dado que existen dos posibilidades de spin. En ocasiones, las alternativas que puede tomar un estado son infinitas y no numerables, por ejemplo, cuando se requiere saber la posición o momento de una partícula.

Un **operador**  $\hat{a}$  en mecánica cuántica es una entidad que actúa sobre los estados del sistema, produciendo típicamente un estado diferente al inicial, pero que sigue perteneciendo al mismo espacio de estados.

$$\hat{a} \cdot (|\psi\rangle) = \hat{a} |\psi\rangle$$

Para cada operador  $\hat{a}$  existen kets “especiales” llamados **eigenkets** o **eigenestados** de  $\hat{a}$  denotados por  $|a'\rangle$ , que al aplicarles el operador resulta el mismo estado  $|a'\rangle$  multiplicado por  $a'$ , es decir:

$$\hat{a} |a'\rangle = a' |a'\rangle$$

en donde  $a'$  puede ser un número real o complejo, y es llamado **eigenvalor** de  $\hat{a}$ .

El espacio dual al espacio ket es llamado **espacio bra**. En el formalismo de la mecánica cuántica se postula que para cada ket  $|\psi\rangle$  existe un bra denotado por  $\langle\psi|$  que pertenece al espacio bra, este espacio es generado por los **eigenbras**  $\langle a'|$  correspondientes a los eigenkets  $|a'\rangle$ .

El producto interno de un bra y un ket se define como:

$$\langle\phi|\psi\rangle = (\langle\phi|) \cdot (|\psi\rangle)$$

y es en general un número complejo. Se postulan dos propiedades fundamentales del producto interno:

$$\langle\phi|\psi\rangle = \langle\phi|\psi\rangle^*$$

$$\langle\psi|\psi\rangle \geq 0$$

en donde  $*$  representa el complejo conjugado, y la igualdad en la segunda propiedad se da solo si  $|\psi\rangle$  es el ket nulo o cero.

Cuando los eigenkets de  $\hat{a}$  están normalizados, es decir:  $\langle a''|a'\rangle = \delta(a'' - a')$  se dice que forman una **base ortonormal** completa para el espacio ket, así, cualquier estado  $|\psi\rangle$  puede ser escrito como:

$$|\psi\rangle = \sum_{a'} c_{a'} |a'\rangle \quad (2.1)$$

multiplicando  $|a''\rangle$  por el lado izquierdo en ambos lados se sigue que:

$$c_{a'} = \langle a'|\psi\rangle$$

sustituyendo en la [Ecuación 2.1](#) se tiene que:

$$|\psi\rangle = \sum_{a'} |a'\rangle \langle a'|\psi\rangle \quad (2.2)$$

dado que  $|\psi\rangle$  es un estado arbitrario, se sigue que:

$$\sum_{a'} |a'\rangle \langle a'| = \hat{1} \quad (2.3)$$

donde  $\hat{1}$  es el operador identidad. La [Ecuación 2.3](#) es conocida como la **relación de completitud**.

Un **observable** en mecánica cuántica puede ser la posición, el momento o el spin

de una partícula, y se puede representar mediante un operador. Una propiedad importante de los observables es que son operadores Hermitianos. Un **operador Hermitiano**  $\hat{x}$  es aquel que cumple:

$$\hat{x} = \hat{x}^\dagger$$

donde  $\hat{x}^\dagger$  es el operador adjunto en el espacio bra, es decir, el dual correspondiente a  $\hat{x}$ . De la definición se sigue que los eigenvalores de un observable son valores reales, y además sus eigenestados son ortonormales, y, por construcción, forman una base completa del espacio ket.

## 2.2 REPRESENTACIÓN MATRICIAL DE UN OPERADOR

Dada una base conformada por los eigenestados  $|a'\rangle$ , por la relación de complejidad un operador  $\hat{x}$  en el espacio ket se puede representar como:

$$\hat{x} = \sum_{a''} \sum_{a'} |a''\rangle \langle a''| \hat{x} |a'\rangle \langle a'| \quad (2.4)$$

en donde hay  $N^2$  elementos, donde  $N$  es la dimensión del espacio ket. Estos elementos se pueden colocar de manera ordenada en una matriz de  $N \times N$ , que puede escribirse explícitamente como:

$$\hat{x} \equiv \begin{pmatrix} \langle a^1 | \hat{x} | a^1 \rangle & \langle a^1 | \hat{x} | a^2 \rangle & \dots \\ \langle a^2 | \hat{x} | a^1 \rangle & \langle a^2 | \hat{x} | a^2 \rangle & \dots \\ \vdots & \vdots & \ddots \end{pmatrix} \quad (2.5)$$

A menudo esta representación de los operadores como matrices es útil cuando se trata de resolver problemas en mecánica cuántica. En este trabajo se utiliza esta representación matricial para escribir al operador Hamiltoniano en los capítulos siguientes.

## 2.3 OPERADOR DE PROPAGACIÓN: EVOLUCIÓN TEMPORAL DE UN SISTEMA

Dado un sistema cuántico a un tiempo inicial  $t_0$ , representado por el estado:

$$|\psi_{t_0}\rangle$$

la evolución temporal del estado a un tiempo  $t > t_0$  se puede escribir como:

$$|\psi(t)\rangle$$

Bajo el formalismo de la mecánica cuántica, se puede visualizar a este nuevo estado  $|\psi(t)\rangle$ , como el resultado de aplicar un operador al estado  $|\psi_{t_0}\rangle$ :

$$|\psi(t)\rangle = \mathcal{U}(t_0, t) |\psi_{t_0}\rangle$$

donde el operador  $\mathcal{U}$  es el **operador de propagación**. Para que este operador sea físicamente consistente, debe satisfacer las siguientes propiedades:

- $\mathcal{U}$  debe ser un operador unitario, es decir, se debe cumplir que:

$$\mathcal{U}^\dagger(t_0, t)\mathcal{U}(t_0, t) = 1$$

- Propiedad de composición, para  $t_0 < t_1 < t_2$ :

$$\mathcal{U}(t_0, t_2) = \mathcal{U}(t_1, t_2)\mathcal{U}(t_0, t_1)$$

A partir de estas propiedades, de la relación de energía Planck-Einstein:  $E = \hbar\omega$ , y de considerar al Hamiltoniano como el generador de la evolución temporal de un sistema físico [27], se puede escribir la siguiente ecuación diferencial para el operador de propagación:

$$i\hbar \frac{\partial}{\partial t} \mathcal{U}(t_0, t) = H \mathcal{U}(t_0, t) \quad (2.6)$$

Se sigue que, para un estado  $|\psi_t\rangle$ , la evolución temporal está dada por la **ecuación de Schrödinger dependiente del tiempo** para un estado: [30]

$$i\hbar \frac{\partial}{\partial t} \mathcal{U}(t_0, t) |\psi_t\rangle = H \mathcal{U}(t_0, t) |\psi_t\rangle \quad (2.7)$$

**AFIRMACIÓN:** Cuando el hamiltoniano  $H$  no tiene dependencia temporal, el operador de propagación toma la siguiente forma:

$$\mathcal{U}(t_0, t) = \exp\left\{\frac{-iH(t - t_0)}{\hbar}\right\} \quad (2.8)$$

Para probar la afirmación anterior, se escribe la función exponencial como serie de potencias:

$$\exp\left\{\frac{-iH(t - t_0)}{\hbar}\right\} = 1 + \frac{-iH(t - t_0)}{\hbar} + \left(\frac{-i^2}{2}\right) \left(\frac{H(t - t_0)}{\hbar}\right)^2 + \dots$$

al derivar respecto al tiempo se obtiene:

$$\frac{\partial}{\partial t} \exp\left\{\frac{-iH(t - t_0)}{\hbar}\right\} = \frac{-iH \cdot 1}{\hbar} + \left(\frac{-i^2}{2}\right) \cdot 2 \left(\frac{H}{\hbar}\right)^2 (t - t_0) + \dots$$

si se multiplica esta última ecuación se observa que se cumple la [Ecuación 2.6](#).

Cuando el Hamiltoniano tiene dependencia temporal, pero commuta en diferentes tiempos, es decir:

$$[H(t_1), H(t_2)] = H(t_1)H(t_2) - H(t_2)H(t_1) = 0$$

para  $t_1 \neq t_2$ , se puede probar que el operador de propagación está dado por:

$$\mathcal{U}(t_0, t) = \exp\left\{-\left(\frac{i}{\hbar}\right) \int_{t_1}^{t_2} H(t') dt'\right\}$$

Encontrar de manera analítica el operador de propagación se vuelve más complejo y complicado cuando el Hamiltoniano tiene dependencia temporal y no commuta en diferentes tiempos, para estos casos se pueden desarrollar métodos de aproximación a la Ecuación 2.7 como: la Expansión de Magnus, Teoría de Perturbaciones con dependencia temporal, integral de caminos, entre otros. [2] [25]

#### 2.4 FUNCIÓN DE ONDA EN EL ESPACIO DE POSICIONES

Un observable comúnmente estudiado en sistemas cuánticos es la posición:  $\hat{x}$ . Como se mencionó en la sección § 2.1, un observable da lugar a un conjunto de eigenvectores:  $\{|x'\rangle, |x''\rangle \dots\}$ , con las siguientes propiedades:

$$\begin{aligned}\hat{x}|x'\rangle &= x' |x'\rangle \\ \langle x''|x'\rangle &= \delta(x'' - x')\end{aligned}$$

Cualquier estado al tiempo  $t$ :  $|\psi(t)\rangle$ , puede ser escrito en función de los eigenestados de la siguiente manera:

$$|\psi(t)\rangle = \int dx |x\rangle \langle x|\psi(t)\rangle$$

En el formalismo de Dirac, el producto interno:

$$\langle x|\psi(t)\rangle = \psi(x, t)$$

es conocido como la **función de onda** para el estado  $|\psi(t)\rangle$ . Las funciones de onda en física cuántica deben cumplir con la condición de normalización:

$$\int_{-\infty}^{\infty} |\psi(x, t)|^2 dx < \infty$$

este conjunto de funciones de onda forma un espacio de Hilbert.

### 2.4.1 Densidad de probabilidad

La **densidad de probabilidad** está definida por la siguiente función:

$$\rho(x) = |\psi(x)|^2 \quad (2.9)$$

Esta relación describe para cada  $x$  en el dominio la probabilidad de encontrar a la partícula en esa posición. La probabilidad de encontrar una partícula en un intervalo  $dx$  alrededor de  $x$  está dado por:

$$|\psi(x)|^2 dx$$

El elemento  $dx$  puede ser un intervalo de longitud, cuando se considera una sola dimensión, o un elemento de volumen  $d^3\vec{x}$ , cuando se tratan las tres dimensiones espaciales. Experimentalmente, está relacionado con la capacidad que tiene un detector (de posición) de asegurar que la partícula está en la vecindad  $dx$  alrededor de  $x$ .

# 3



## REPRESENTACIÓN DE VARIABLE DISCRETA

Existen diferentes métodos numéricos para resolver la ecuación de Schrödinger dependiente del tiempo ([Ecuación 2.7](#)). En esta capítulo se desarrollará un método particular conocido como el método de **representación de variable discreta**, **DVR** por sus siglas en inglés.

Formalmente, el espacio de Hilbert de las funciones de onda es infinito, sin embargo, para resolver y tratar problemas numéricamente, es necesario hacer un truncamiento de la dimensión del espacio a un número finito  $N$ . Este espacio tiene el mismo formalismo de la mecánica cuántica, pues la dimensión del espacio de Hilbert está dado por las distintas alternativas que puede tomar el sistema físico en cuestión, de manera que este espacio de dimensión  $N$  puede ser un espacio completo para otro problema en mecánica cuántica. Por ejemplo, si  $N = 2$  el espacio generado puede verse como el espacio de Hilbert asociado al problema de la medición del spin en los átomos de plata. [\[9\]](#)

### 3.1 PROYECCIÓN ESPECTRAL Y COLOCACIÓN

Una función de onda y sus operadores se pueden representar mediante una base de funciones ortogonales, a esta base se le conoce como **base espectral**:

$$\{\phi_i(x)\}_{i=1}^N$$

que, por ser funciones ortogonales cumplen que:

$$\langle \phi_i(x) | \phi_j(x) \rangle = \delta_{ij}$$

Como se mencionó anteriormente, la dimensión del espacio de Hilbert se debe reducir a un número finito  $N$ , esta reducción de dimensión se puede expresar mediante un **operador de proyección**:

$$P_N = \sum_{n=1}^N |\phi_n\rangle \langle \phi_n| \tag{3.1}$$

Mediante el operador  $P_N$  se puede mapear la dinámica del espacio de Hilbert al espacio reducido de Hilbert, en particular, es de interés para resolver la [Ecuación 2.7](#) conocer cómo se representa el operador Hamiltoniano en su representación matricial:

$$H = \frac{\hat{p}^2}{2m} + V(\hat{x}) \quad (3.2)$$

Asumiendo que la matriz de elementos  $\langle \phi_m | H | \phi_n \rangle$  es conocida dada la base espectral, el Hamiltoniano en el espacio reducido de Hilbert se puede representar como:

$$H_N = P_N H P_N \quad (3.3)$$

La ecuación de Schrödinger dependiente del tiempo:

$$i\hbar \frac{\partial \psi}{\partial t} = H\psi \quad (3.4)$$

puede escribirse como un conjunto de dos ecuaciones diferenciales acopladas a partir de las siguientes definiciones: [30]

$$Q_N = \mathbb{1} - P_N \quad (3.5)$$

$$\psi_N = P_N \psi \quad (3.6)$$

$$\psi_{perp} = Q_N \psi = \psi - \psi_N \quad (3.7)$$

de la [Ecuación 3.7](#) se sigue que:  $\psi = \psi_N + \psi_{\perp}$ , sustituyendo en la [Ecuación 3.4](#) se tiene:

$$i\hbar \frac{\partial(\psi_N + \psi_{\perp})}{\partial t} = H(\psi_N + \psi_{\perp}) \quad (3.8)$$

multiplicando por el operador  $P_N$  por la izquierda en ambos lados se obtiene:

$$i\hbar \frac{\partial(P_N \psi_N + P_N \psi_{\perp})}{\partial t} = P_N H(\psi_N + \psi_{\perp}) \quad (3.9)$$

por definición  $P_N \psi_{\perp} = P_N Q_N \psi = P_N (\mathbb{1} - P_N) \psi = (P_N - P_N^2) \psi = (P_N - P_N) \psi = 0$ , pues  $P_N$  es un operador de proyección, también se sigue que  $P_N \psi_N = P_N P_N \psi = P_N^2 \psi = P_N \psi = \psi_N$  De esta manera la [Ecuación 3.9](#) se convierte en:

$$i\hbar \frac{\partial \psi_N}{\partial t} = P_N H(\psi_N + \psi_{\perp}) \quad (3.10)$$

multiplicando por  $\mathbb{1} = P_N + Q_N$  en el lado derecho de la ecuación se sigue:

$$i\hbar \frac{\partial \psi_N}{\partial t} = P_N H(P_N + Q_N)(\psi_N + \psi_{\perp}) \quad (3.11)$$

$$= (P_N H P_N + P_N H Q_N)(\psi_N + \psi_{\perp}) \quad (3.12)$$

$$= P_N H P_N \psi_N + P_N H Q_N \psi_N + P_N H Q_N \psi_{\perp} + P_N H Q_N \psi_{\perp} \quad (3.13)$$

como se mostró anteriormente  $P_N \psi_{\perp} = 0$ , además  $Q_N \psi_N = (\mathbb{1} - P_N) \psi_N = \psi_N - P_N \psi_N = \psi_N - \psi_N = 0$ , de esta manera el segundo y tercer término del lado derecho en la Ecuación 3.11 se hacen 0 y se sigue que:

$$i\hbar \frac{\partial \psi_N}{\partial t} = P_N H P_N \psi_N + P_N H Q_N \psi_{\perp} \quad (3.14)$$

Por otro lado, multiplicando el operador  $Q_N$  por la izquierda en ambos lados de la Ecuación 3.8 se tiene:

$$i\hbar \frac{\partial(Q_N \psi_N + Q_N \psi_{\perp})}{\partial t} = Q_N H(\psi_N + \psi_{\perp}) \quad (3.15)$$

$$i\hbar \frac{\partial \psi_{\perp}}{\partial t} = Q_N H(\psi_N + \psi_{\perp}) \quad (3.16)$$

multiplicando por  $\mathbb{1} = P_N + Q_N$  en el lado derecho de la ecuación se sigue:

$$i\hbar \frac{\partial \psi_{\perp}}{\partial t} = Q_N H(P_N + Q_N)(\psi_N + \psi_{\perp}) \quad (3.17)$$

$$= (Q_N H P_N + Q_N H Q_N)(\psi_N + \psi_{\perp}) \quad (3.18)$$

$$= Q_N H P_N \psi_N + Q_N H Q_N \psi_N + Q_N H P_N \psi_{\perp} + Q_N H Q_N \psi_{\perp} \quad (3.19)$$

por los argumentos anteriores, el segundo y tercer término del lado derecho de la Ecuación 3.17 se hacen 0, obteniendo:

$$i\hbar \frac{\partial \psi_{\perp}}{\partial t} = Q_N H P_N \psi_N + Q_N H Q_N \psi_{\perp} \quad (3.20)$$

de esta manera la Ecuación 3.14 y la Ecuación 3.20 son un conjunto de dos ecuaciones diferenciales acopladas.

Para este trabajo se usará la aproximación de Galerkin [11], en donde se desprecia la contribución de  $\psi_{\perp}$ , de esta forma, la TDSE a resolver es:

$$i\hbar \frac{\partial \psi_N}{\partial t} = P_N H P_N \psi_N \quad (3.21)$$

### 3.1.1 Representación de la función de onda bajo el operador de proyección

Una función de onda  $\psi(x)$  se puede escribir como una suma infinita en términos de funciones ortonormales  $\phi_i$ :

$$\psi(x) = \sum_{n=1}^{\infty} a_n \phi_n(x) \quad (3.22)$$

con:

$$\int \phi_m^*(x) \phi_n(x) dx = \delta_{mn}$$

$$a_n = \int \phi_n^*(x) \psi(x) dx$$

para  $m, n = 1, 2, 3, \dots, \infty$ .

Utilizando la definición del operador de proyección de la Ecuación 3.1, se puede probar que:

$$\psi_N(x) = P_N \psi(x) = \sum_{n=1}^N a_n \phi_n(x) \quad (3.23)$$

### 3.1.2 Definición del proyector de colocación

Dado un conjunto de puntos en el espacio de posiciones:  $\{x_i\}$  con  $i = 1, 2, 3, \dots, N$ , la siguiente relación:

$$\psi_N(x_i) = P_N \psi(x_i) = \sum_{n=1}^N b_n \phi_n(x_i) = \psi(x_i) \quad (3.24)$$

está asociada a la **colocación**<sup>1</sup> del operador de proyección, y los puntos  $\{x_i\}$  son llamados **puntos de colocación**. Los coeficientes  $b_n$  están determinados por la condición de que:  $\psi_N(x) = \psi(x)$  en el conjunto de puntos de colocación.

Es importante notar que los coeficientes  $b_n$  son en general distintos a los coeficientes  $a_n$  de la Ecuación 3.23. A pesar de que el operador  $P_N$  en la proyección espectral general el mismo espacio reducido de Hilbert que el operador  $P_N$  en la colocación, estos operadores difieren en el estado que producen en el mismo espacio. Si se define:

$$\chi = \psi - \psi_N \quad (3.25)$$

Para la proyección espectral se tiene que:

$$\chi(x) = \sum_{n=1}^{\infty} a_n \phi_n(x) - \sum_{n=1}^N a_n \phi_n(x) = \sum_{n=N+1}^{\infty} a_n \phi_n(x) \quad (3.26)$$

---

<sup>1</sup>Los **métodos de colocación** son soluciones numéricas de un conjunto de ecuaciones, cuya solución resulta ser exacta en un conjunto discreto de puntos llamados puntos de colocación.[30]

de manera que:

$$\langle \chi | \psi_N \rangle = \int \left( \sum_{n=N+1}^{\infty} a_n^* \phi_n^*(x) \right) \left( \sum_{m=1}^N a_m \phi_m(x) \right) dx = 0 \quad (3.27)$$

lo que muestra que el proyector espectral es un proyector ortogonal. Por otro lado, si se considera la colocación del proyector:

$$\chi(x) = \psi - \psi_N = \sum_{n=1}^{\infty} a_n \phi_n(x) - \sum_{n=1}^N b_n \phi_n(x) \quad (3.28)$$

el producto está dado por:

$$\langle \chi | \psi_N \rangle = \int \left( \sum_{n=1}^{\infty} a_n^* \phi_n^*(x) - \sum_{n=1}^N b_n^* \phi_n^*(x) \right) \left( \sum_{m=1}^N b_m \phi_m(x) \right) dx \neq 0 \quad (3.29)$$

mostrando así que el proyector de colocación no es un proyector ortogonal. La Figura 3.1 muestra una interpretación geométrica de lo anterior, en este caso el espacio reducido es  $\mathbb{R}^2$ , y el vector  $(x, y, z) \in \mathbb{R}^3$  es proyectado por dos distintos operadores: uno ortogonal y uno no ortogonal, como se observa, el vector proyectado en ambos casos pertenece al mismo espacio reducido, pero no son el mismo vector. [30]

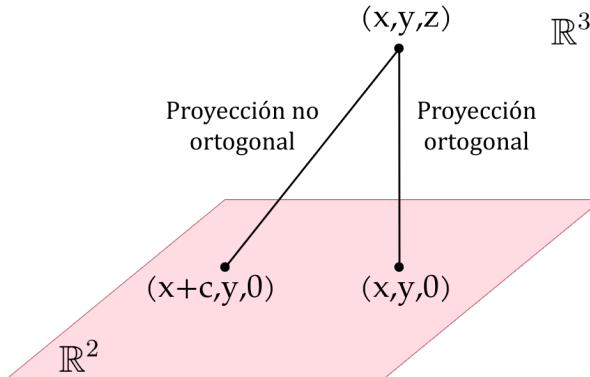


Figura 3.1: Interpretación geométrica de una proyección ortogonal y una no ortogonal del un vector en el espacio  $\mathbb{R}^3$  al espacio  $\mathbb{R}^2$ .

### 3.1.3 Colocación ortogonal

Un proyector que cumpla con las condiciones de colocación y además sea un proyector ortogonal debe tener funciones  $\phi_n(x)$  que satisfagan las relaciones discretas de ortogonalidad definidas en los puntos de colocación: [30]

$$\sum_{j=1}^N \phi_m^*(x_j) \phi_n(x_j) \Delta_j = \delta_{mn}, \quad m, n = 1, \dots, N \quad (3.30)$$

en donde  $\Delta_j$  es un factor de peso. Los coeficientes  $b_n$  de la Ecuación 3.24 pueden encontrarse a partir de la Ecuación 3.30 multiplicando a la izquierda por  $\phi_m^*(x_j) \Delta_j$  y sumando sobre  $j$ :

$$b_n = \sum_{j=1}^N \psi(x_j) \phi_n^*(x_j) \Delta_j \quad (3.31)$$

Lo anterior es una buena aproximación discreta a la relación de los coeficientes  $a_n$  en la Ecuación 3.22, que cumplen la condición para una proyección ortogonal. Los esquemas que satisfacen la Ecuación 3.30 son llamados **esquemas de colocación ortogonales**, y pueden ser redefinidos de manera más sencilla de la siguiente manera:

$$\Phi_n(x_j) \equiv \sqrt{\Delta_j} \phi_n(x_j) \quad (3.32)$$

así, la Ecuación 3.30 puede escribirse como:

$$\sum_{j=1}^N \Phi_m^*(x_j) \Phi_n(x_j) = \delta_{mn}, \quad m, n = 1, \dots, N \quad (3.33)$$

La ecuación anterior puede escribirse en forma matricial como:

$$\Phi^\dagger \Phi = \mathbb{1} \quad (3.34)$$

lo que indica que la transformación  $\Phi$  es unitaria, por lo que se tiene también la siguiente relación:

$$\Phi \Phi^\dagger = \mathbb{1} \quad (3.35)$$

que puede escribirse de manera extendida como:

$$\sum_{n=1}^N \Phi_n(x_i) \Phi_n^*(x_j) = \delta_{ij}, \quad i, j = 1, \dots, N \quad (3.36)$$

es importante notar que la ecuación anterior es distinta a la Ecuación 3.33, pues la ecuación Ecuación 3.36 implica una relación de ortogonalidad para los diferentes puntos en la malla del espacio de posiciones. La matriz  $\Phi$  será llamada matriz de colocación ortogonal.

### 3.2 BASE PSEUDO-ESPECTRAL

Una **base pseudo-espectral**:  $\{\theta_j\}_{j=1}^N$  se define como la base de las funciones localizadas espacialmente. La base de funciones ortogonales  $\{\phi_n\}_{n=1}^N$ , la colocación

en la Ecuación 3.24, los puntos de colocación  $\{x_i\}_{i=1}^N$ , y los factores de peso definidos como  $\Delta_j$  con  $j = 1, \dots, N$ , determinan completamente la forma de la base pseudo-espectral.

Reemplazando  $\Phi_n(x_i)$  por  $\phi_n(x)$  en el primer factor de la Ecuación 3.36 se definen las funciones pseudo-espectrales base:

$$\theta_j(x) \equiv \sum_{n=1}^N \phi_n(x) \Phi_n^*(x_j) \quad (3.37)$$

Las funciones  $\{\theta_j\}$  son localizadas, cada una alrededor de los diferentes valores  $x_j$ . De la Ecuación 3.37, Ecuación 3.32 y Ecuación 3.36 se tiene que las funciones satisfacen:

$$\theta_j(x_i) = \Delta_j^{-1/2} \delta_{ij} \quad (3.38)$$

La Ecuación 3.37 puede escribirse en notación matricial como:

$$\Phi^\dagger \phi(x) = \theta(x) \quad (3.39)$$

de manera que  $\{\theta_j\}$  y  $\{\phi_n\}$  son bases que están relacionadas mediante una transformación unitaria  $\Phi^\dagger$  y generan el mismo espacio de Hilbert reducido. Como consecuencia, el operador de proyección es idéntico en ambas bases:

$$P_N = \sum_{n=1}^N |\phi_n\rangle \langle \phi_n| = \sum_{j=1}^N |\theta_j\rangle \langle \theta_j| \quad (3.40)$$

### 3.2.1 Completes y ortogonalidad de las funciones base pseudo-espectrales

La base de las funciones localizadas  $\{\theta_j\}$  cumple las propiedades de ortogonalidad y completos, para mostrar esto se multiplica por  $\phi_m^*(x)$  ambos lados de la ecuación Ecuación 3.37 y se integra sobre el dominio:

$$\begin{aligned} \int \phi_m^*(x) \theta_j(x_j) dx &= \int \sum_{n=1}^N \phi_m^*(x) \phi_n(x) \Phi_n^*(x_j) dx \\ \langle \phi_m | \theta_j \rangle &= \Phi_n^*(x_j) \end{aligned} \quad (3.41)$$

De esta manera, la relación de ortogonalidad es:

$$\sum_{n=1}^N \Phi_n(x_i) \Phi_n^*(x_j) = \delta_{ij} = \sum_{n=1}^N \langle \theta_i | \phi_n \rangle \langle \phi_n | \theta_j \rangle = \langle \theta_i | \theta_j \rangle \quad (3.42)$$

mientras que la de completos está dada por:

$$\sum_{j=1}^N \Phi_m(x_j) \Phi_n^*(x_j) = \delta_{mn} = \sum_{j=1}^N \langle \phi_m | \theta_j \rangle \langle \theta_j | \phi_n \rangle \quad (3.43)$$

La Ecuación 3.42 significa que las funciones  $\theta_j$  tienen un pico en el valor correspondiente a  $x_j$  mientras que se desvanece en cualquier otro punto de la malla  $x_i$ . La Ecuación 3.43 es una relación de completos respecto a la suma sobre todos los puntos de la malla.

### 3.2.2 El proyector de colocación en la base pseudo-espectral

En la sección § 3.1.2 se definió el proyector de colocación utilizando la base espectral  $\{\phi_i(x)\}_{i=1}^N$ , en esta sección se escribirá la relación de colocación en la base pseudo-espectral.

La Ecuación 3.37 puede invertirse usando la matriz de transformación unitaria  $\Phi$ :

$$\phi_n(x) = \sum_{i=1}^N \Phi_n(x_i) \theta_i(x) \quad (3.44)$$

usando esta relación, la Ecuación 3.24 y la Ecuación 3.32, se puede reescribir la relación de colocación como:

$$\psi_N(x) = \sum_{n=1}^N b_n \phi_n(x) = \sum_{i=1}^N \left\{ \sum_{n=1}^N b_n \Phi_n(x_i) \right\} \theta_i(x) \quad (3.45)$$

$$\psi_N(x) = \sum_{i=1}^N \psi(x_i) \theta_i(x_i) \Delta_i^{1/2} \quad (3.46)$$

En los puntos de colocación  $\{x_i\}$  se tiene:

$$\psi_N(x_i) = \sum_{i=1}^N \psi(x_i) \theta_i(x_i) \Delta_i^{1/2} \quad (3.47)$$

de esta manera se tiene la relación de colocación en la base pseudo-espectral  $\{\theta_i(x)\}_{i=1}^N$ .

## 3.3 ALGORITMO DVR APLICADO A UN PROCESO FÍSICO-QUÍMICO

En esta sección se aplicarán los conceptos revisados a lo largo del capítulo para resolver un problema físico-químico que involucra potenciales dependientes del tiempo utilizando el método **DVR**.

La implementación numérica se realizó en Python 3.9.7 y se encuentra disponible en el repositorio:  Transferencia de Protones

### 3.3.1 Sistema de transferencia de protones

Los sistemas de **transferencia de protones** ocurren en un complejo de enlaces de hidrógeno:  $A-H \cdots A'$ . El modelo simplificado se muestra en la siguiente figura:

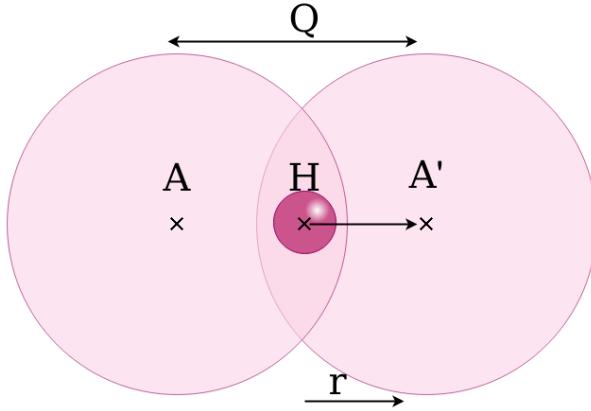


Figura 3.2: Descripción del modelo de transferencia de protones  $H^+$ .

En donde la coordenada  $Q$  hace referencia a la separación entre los átomos  $A$  y  $A'$ , mientras que la coordenada  $r$  es la distancia del protón al centro de los enlaces. [5].

En las descripciones teóricas de la transferencia de protones, a menudo el hidrógeno es representado moviéndose a través de un pozo de doble potencial unidimensional. [17]

A continuación se presenta un modelo particular de potencial para el sistema de transferencia de protones, en donde la descripción está dada por la coordenada del protón  $r \in [-1.5 \text{ \AA}, 1.5 \text{ \AA}]$ . Los sistemas de transferencia de protones son, en general, electrónicamente adiabáticos, en el sentido de que ocurren en el estado base o ground state (GS) en inglés, y no involucran los estados excitados del sistema o excited states (ES) en inglés. [17].

A cada tiempo  $t$  el valor del potencial  $V(r, t)$  está dado por el estado base, que es el eigenvalor más bajo de la siguiente matriz:

$$\begin{pmatrix} U_1(r, R(t)) & V \\ V & U_2(r, R(t)) + X(t) \end{pmatrix} \quad (3.48)$$

En donde  $U_1(r, R(t))$  y  $U_2(r, R(t))$  son potenciales de oscilador armónico, y  $V$  es una constante de acoplamiento electrónico.

$$U_1(r, R(t)) = \frac{1}{2}m\omega_1^2 \left( r + \frac{R(t)}{2} \right) \quad (3.49)$$

$$U_2(r, R(t)) = \frac{1}{2}m\omega_2^2 \left( r - \frac{R(t)}{2} \right) \quad (3.50)$$

En las ecuaciones de potenciales de oscilador armónico,  $m$  se refiere a la masa del protón,  $\omega_1$  y  $\omega_2$  son las frecuencias del potencial armónico del protón. Los términos  $U_1(r, R(t))$  y  $U_2(r, R(t))$  están desplazados mediante un término de energía dependiente del tiempo  $X(t)$ , y un término de distancia dependiente del tiempo  $R(t)$ . La dinámica de  $X(t)$  corresponde a las fluctuaciones del entorno, mientras que  $R(t)$  representa las vibraciones de los sitios donantes y aceptores de protones: [29]

$$X(t) = \lambda \cos(\omega_x t + \theta_x) + X_{eq} \quad (3.51)$$

$$R(t) = (R_0 - R_{eq}) \cos(\omega_R t + \theta_R) + R_{eq} \quad (3.52)$$

En donde  $\lambda$  es la amplitud del sesgo de energía,  $\omega_x$  es la frecuencia de las oscilaciones de polarización de energía,  $\theta_x$  es una fase inicial aleatoria;  $R_{eq}$  es la distancia de equilibrio entre los mínimos de los potenciales armónicos,  $R_0$  es el desplazamiento inicial desde el equilibrio,  $\omega_R$  es la frecuencia de las oscilaciones de la distancia donante-aceptor de protones y  $\theta_R$  es una fase inicial aleatoria. [29]

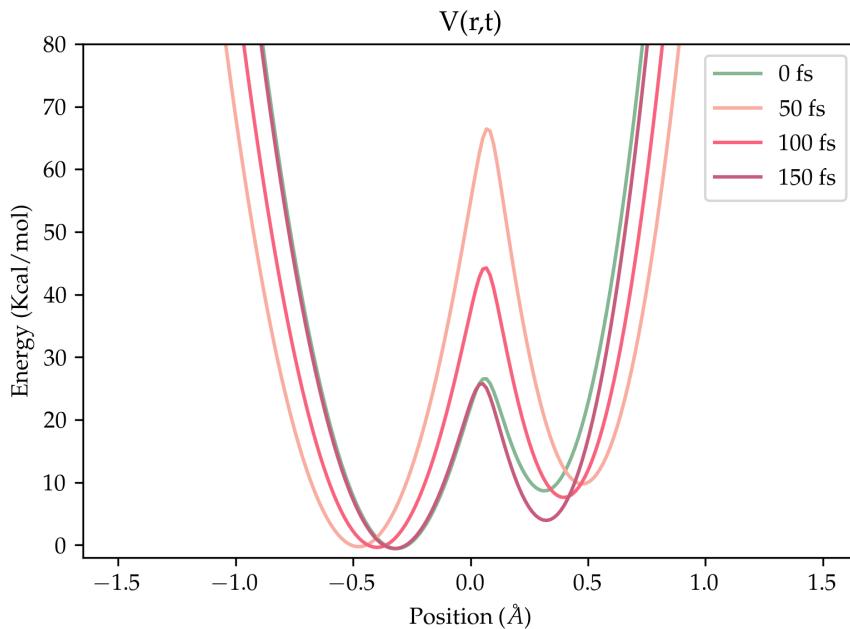


Figura 3.3: Potencial  $V(r, t)$  a diferentes tiempos.

La Figura 3.3 muestra un ejemplo de potencial  $V(r, t)$  generado con la implementación de las ecuaciones anteriores. Los parámetros de potencial utilizados se

muestran en la [Tabla A.1](#)

La [Figura 3.4](#) muestra el estado base que se toma como el potencial, el primer estado excitado del sistema correspondiente al eigenvalor más grande de la matriz en la [Ecuación 3.48](#), y los potenciales de oscilador armónico [Ecuación 3.49](#) y [Ecuación 3.50](#) a un tiempo  $t$ .

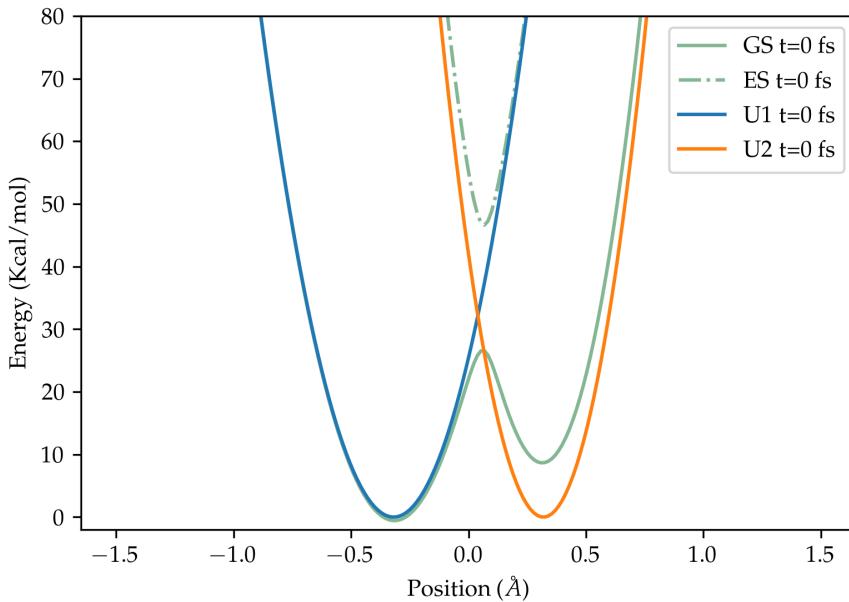


Figura 3.4: Estado base (GS), primer estado excitado (ES) y potenciales armónicos  $U_1(r, (R(t)))$  y  $U_2(r, R(t))$  al tiempo  $t = 0$ .

### 3.3.2 Elección de una base ortonormal y malla

Para proceder con la solución a la [TDSE Ecuación 2.7](#), elegimos una base espectral de funciones ortogonales: [4]

$$\phi_n(r) = \sqrt{\frac{2}{b-a}} \sin\left(\frac{n\pi(r-a)}{b-a}\right) \quad n = 1, \dots, N \quad (3.53)$$

y una malla en el espacio de posiciones:

$$r_i = a + \frac{(b-a)i}{N+1} \quad i = 0, \dots, N \quad (3.54)$$

donde:  $a = -1.5$ ,  $b = 1.5$  y  $N = 200$  para la implementación numérica, es decir, que el espacio reducido de Hilbert del sistema tiene una dimensión de  $N = 200$ . La [Figura 3.5](#) muestra las primeras cinco funciones ortogonales de la base en la malla

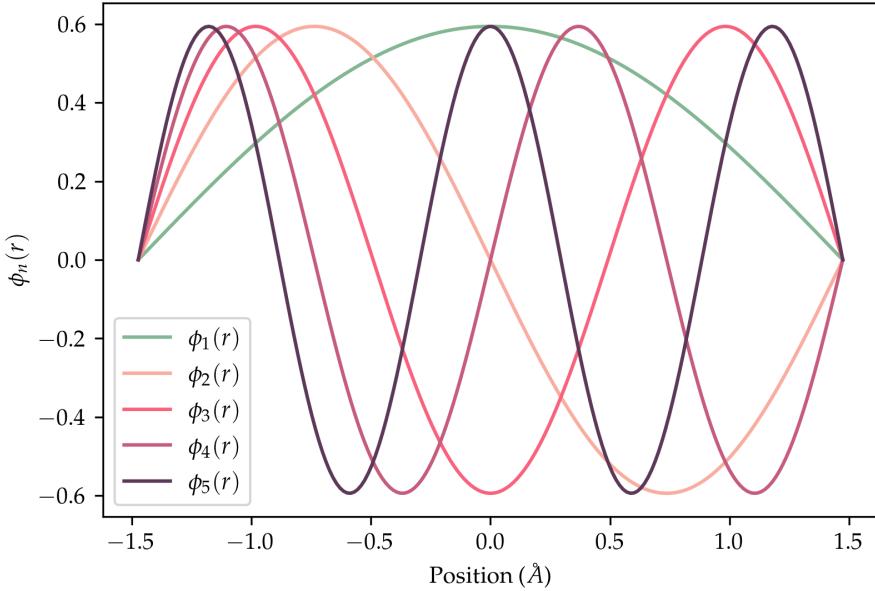


Figura 3.5: Funciones ortogonales  $\phi_n(r)$  en la malla  $\{r_i\}_{i=1}^N$  para  $n = 1, \dots, 5$ .

del espacio de posiciones. Las funciones  $\phi_n$  están construidas para que se cumpla:  $\phi_n(r = r_0 = a) = \phi_n(r = r_{N+1} = b) = 0$ .

Dada esta base, los siguientes elementos de matriz pueden ser calculados de manera exacta:

$$\langle \phi_j | \phi_k \rangle = \delta_{jk} \quad (3.55)$$

$$\langle \phi_j | \frac{d}{dr} | \phi_k \rangle = \text{mod}(j - k, 2) \frac{4}{b - a} \left( \frac{jk}{j^2 - k^2} \right) \text{ con } j \neq k \quad (3.56)$$

$$\langle \phi_j | \frac{d^2}{dr^2} | \phi_k \rangle = - \left( \frac{j\pi}{b - a} \right)^2 \delta_{jk} \quad (3.57)$$

Dado que la energía cinética está dada por  $T = -\frac{\hbar^2}{2m} \frac{d^2}{dr^2}$ , usando la Ecuación 3.57 se tiene que los elementos de matriz de energía cinética en la base espectral están dados por:

$$T_{jk}^\phi = \langle \phi_j | T | \phi_k \rangle = -\frac{\hbar^2}{2m} \langle \phi_j | \frac{d^2}{dr^2} | \phi_k \rangle \quad (3.58)$$

$$T_{jk}^\phi = \frac{\hbar^2}{2m} \left( \frac{j\pi}{b - a} \right)^2 \delta_{jk} \quad (3.59)$$

### 3.3.3 Representación del operador espacial $\hat{r}$ en la base espectral

La matriz que representa al operador espacial  $\hat{r}$  en la base de funciones ortogonales  $\{\phi_n\}_{n=1}^N$  está dada por  $\langle \phi_j | r | \phi_k \rangle$ , sin embargo, para hacer más sencilla la diagonalización en el siguiente paso se usará la siguiente transformación:

$$f(r) = \cos \pi \frac{(r - r_0)}{b - a} \quad (3.60)$$

de esta manera, los elementos de matriz están dados por:

$$F_{jk} = \langle \phi_j | f(r) | \phi_k \rangle = \frac{1}{2} (\delta_{j,k+1} + \delta_{j,k-1}) = \frac{1}{2} \begin{pmatrix} 0 & 1 & 0 & \dots \\ 1 & 0 & 1 & \dots \\ 0 & 1 & 0 & \dots \\ \vdots & \vdots & \vdots & \ddots \end{pmatrix} \quad (3.61)$$

### 3.3.4 Diagonalización mediante una transformación unitaria

La matriz de la [Ecuación 3.61](#) es una matriz simétrica con entradas reales, por el teorema espectral de dimensión finita, puede ser diagonalizada por una matriz ortogonal:

$$\Phi_{j\alpha} = \sqrt{\frac{2}{N+1}} \sin \frac{j\alpha\pi}{N+1} \quad (3.62)$$

que además es unitaria:

$$\{\Phi^\dagger \Phi\}_{jk} = \{\Phi \Phi\}_{jk} \quad (3.63)$$

$$= \frac{2}{N+1} \sum_{\alpha}^N \sin \pi \frac{j\alpha}{N+1} \sin \pi \frac{k\alpha}{N+1} = \begin{cases} 1 & \text{si } j = k \\ 0 & \text{si } j \neq k \end{cases} \quad (3.64)$$

y sus eigenvalores están dados por: [\[Noschese2012\]](#)

$$f_\alpha = \cos \frac{\alpha\pi}{N+1} \quad (3.65)$$

de esta manera, los puntos en la malla están definidos por:

$$\begin{aligned} r_\alpha &= f^{-1}(f_\alpha) = r_0 + \frac{b-a}{\pi} f_\alpha = r_0 + \alpha \frac{b-a}{N+1} \\ r_\alpha &= r_0 + \alpha \Delta r \end{aligned} \quad (3.66)$$

$$\text{con } \alpha = 1, 2, \dots, N \text{ y } \Delta r = \frac{b-a}{N+1}$$

### 3.3.5 Representación del Hamiltoniano en la base pseudo-espectral

La Ecuación 3.62 permite pasar de la base espectral de las funciones ortogonales a la base pseudo-espectral de las funciones localizadas espacialmente. La base de las funciones pseudo-espectrales se puede escribir explícitamente usando la Ecuación 3.37:

$$\theta_\alpha(r) = \sum_{j=1}^N \phi_j(x) \Phi_{j\alpha} \quad (3.67)$$

$$= \sum_{j=1}^N \sqrt{\frac{2}{b-a}} \sin \frac{j\pi(r-r_0)}{b-a} \sqrt{\frac{2}{N+1}} \sin \frac{j\alpha\pi}{N+1} \quad (3.68)$$

$$= \frac{2}{\sqrt{(b-a)(N+1)}} \sum_{j=1}^N \sin \frac{j\pi(r-r_0)}{b-a} \sin \frac{j\alpha\pi}{N+1} \quad (3.69)$$

usando identidades trigonométricas de multiplicación a suma:

$$= \frac{2}{\sqrt{(b-a)(N+1)}} \sum_{j=1}^N \frac{1}{2} \left\{ \cos j\pi \left( \frac{r-r_0}{b-a} - \frac{\alpha}{N+1} \right) - \cos j\pi \left( \frac{r-r_0}{b-a} + \frac{\alpha}{N+1} \right) \right\} \quad (3.70)$$

$$= \frac{1}{\sqrt{(b-a)(n+1)}} \left\{ \sum_{j=1}^N \cos j\pi \left( \frac{r-r_0}{b-a} - \frac{\alpha}{N+1} \right) - \sum_{j=1}^N \cos j\pi \left( \frac{r-r_0}{b-a} + \frac{\alpha}{N+1} \right) \right\} \quad (3.71)$$

de acuerdo a § A.1:

$$\begin{aligned} \theta_\alpha(r) &= \frac{1}{\sqrt{(b-a)(n+1)}} \left\{ \cos \frac{N+1}{2} \pi \left( \frac{r-r_0}{b-a} - \frac{\alpha}{N+1} \right) \frac{\sin \frac{N}{2} \pi \left( \frac{r-r_0}{b-a} - \frac{\alpha}{N+1} \right)}{\sin \frac{1}{2} \pi \left( \frac{r-r_0}{b-a} - \frac{\alpha}{N+1} \right)} \right. \\ &\quad \left. - \cos \frac{N+1}{2} \pi \left( \frac{r-r_0}{b-a} + \frac{\alpha}{N+1} \right) \frac{\sin \frac{N}{2} \pi \left( \frac{r-r_0}{b-a} + \frac{\alpha}{N+1} \right)}{\sin \frac{1}{2} \pi \left( \frac{r-r_0}{b-a} + \frac{\alpha}{N+1} \right)} \right\} \quad (3.72) \end{aligned}$$

La Figura 3.6 muestra algunas funciones base pseudo-espectrales. Estas funciones tienen un valor distinto de cero únicamente en el punto  $r_{i=\alpha}$ , como se esperaba por las propiedades descritas en la sección § 3.2.

La representación de la matriz de energía potencial en la base pseudo-espectral es simple, pues las funciones  $\{\theta_\alpha\}$  son localizadas en el espacio de posiciones y cumplen la condición de ortogonalidad:  $\langle \theta_\alpha | \theta_\beta \rangle = \delta_{\alpha\beta}$ , así:

$$V_{ij}^\theta = \langle \theta_i | V(\hat{r}) | \theta_j \rangle = V(r_i) \delta_{ij} \quad (3.73)$$

$V^\theta \in \mathcal{M}_{200 \times 200}(\mathbb{R})$  es una matriz diagonal.

Dado un tiempo  $t$ , los elementos de la diagonal  $V_{ii}^\theta$  están dados por  $V(r_i, t)$  (Ecu-

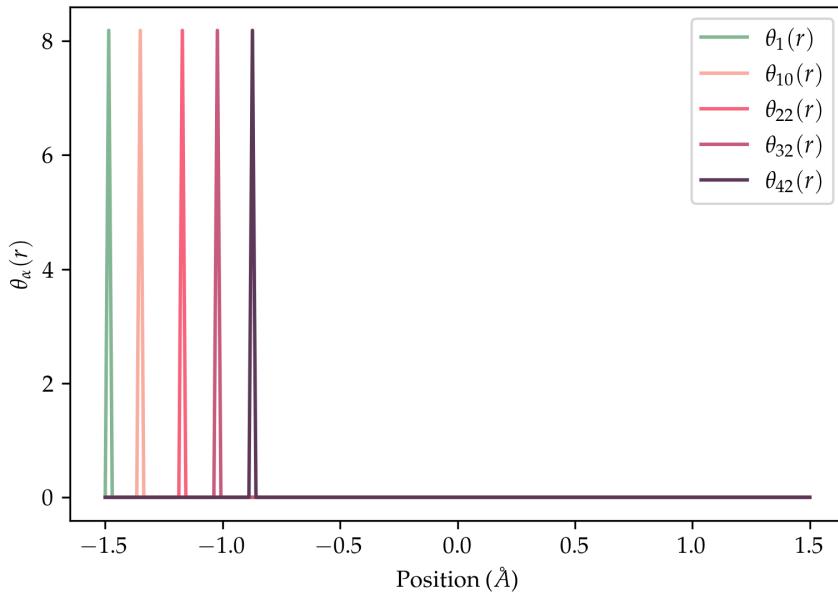


Figura 3.6: Funciones ortogonales  $\theta_\alpha(r)$  en la malla  $\{r_i\}_{i=1}^N$  para  $\alpha = 1, 10, 22, 32, 42$ .

ción 3.48), con  $i = 0, \dots, 199$ .

Escribir directamente la matriz de energía cinética en la base pseudo-espectral es, en general, complicado, por ello comúnmente se calcula en la base espectral y se utiliza la matriz de transformación unitaria Ecuación 3.62 para escribirla en la base pseudo-espectral:

$$T^\theta = \Phi T^\phi \Phi \quad (3.74)$$

en donde  $T^\phi$  está dada por la Ecuación 3.59, de esta manera, la matriz del Hamiltoniano en la base pseudo-espectral al tiempo  $t$  es:

$$H(t)^\theta = V(t)^\theta + T^\theta \quad (3.75)$$

en donde el Hamiltoniano tiene dependencia temporal debido a que el potencial del sistema es dependiente del tiempo.

### 3.3.6 Propagación de un paquete de onda

Sea un paquete de onda al tiempo  $t = 0$  (Figura 3.7):

$$\psi(r, 0) = \sum_{i=1}^N C_i \cdot \theta_i(r) \quad (3.76)$$

con  $C_i$  números complejos de una distribución normal aleatoria alrededor de la malla, elegidos de tal forma que:  $\langle \psi(r, 0) | \psi(r, 0) \rangle = 1$

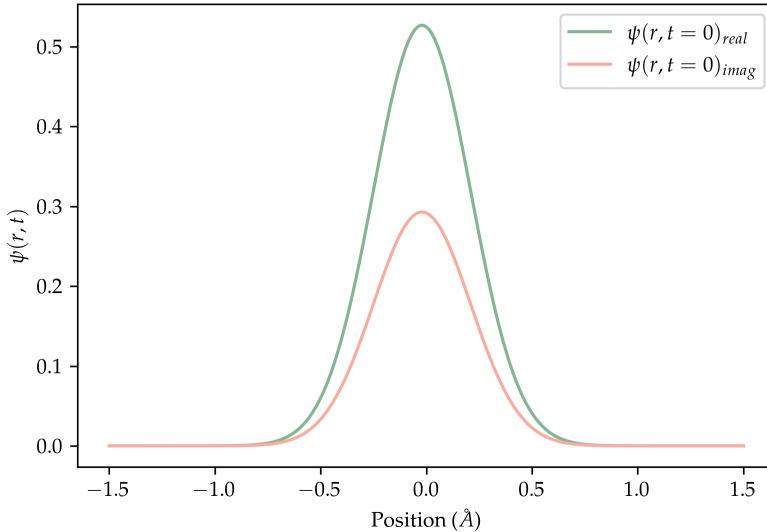


Figura 3.7: Paquete inicial de onda al tiempo  $t = 0$ . Parte real y compleja.

Si se toma un intervalo de tiempo  $\Delta t$  lo suficientemente pequeño como para que el potencial se pueda considerar constante en ese intervalo de tiempo ( $\approx 1 fs$  para el sistema en cuestión), la evolución temporal del paquete de onda está dado por (Ecuación 2.8):

$$\psi(r, t) = \exp\{-iH^\theta t/\hbar\} \psi(r, t - 1) \quad (3.77)$$

Si  $H^\theta = UDU^{-1}$ , con  $D$  una matriz diagonal:

$$\psi(r, t) = \exp\{-iUDU^{-1}t/\hbar\} \psi(r, t - 1)$$

así,

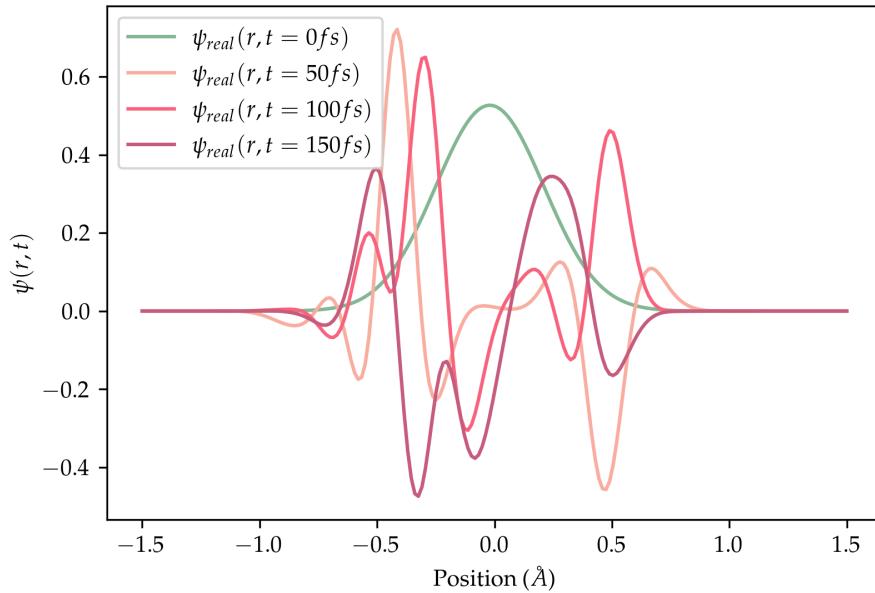
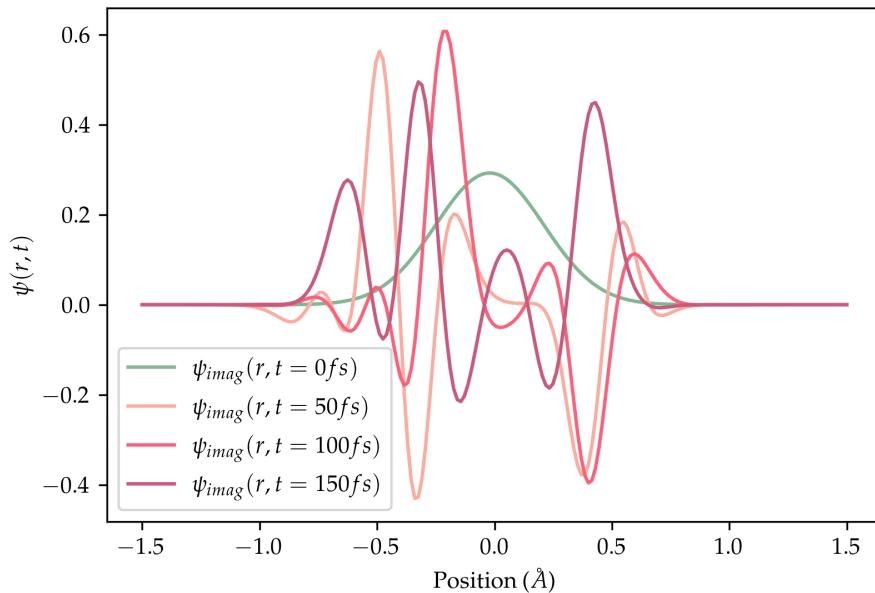
$$\psi(r, t) = U \exp\left\{\frac{-it}{\hbar} D\right\} U^{-1} \psi(r, t - 1) \quad (3.78)$$

En donde la matriz  $U$  está formada por los  $N$  eigenvectores de  $H^\theta$  como vectores columna, y:

$$D = U^{-1} H^\theta U$$

La Figura 3.8 y la Figura 3.9 muestran la evolución de la parte real e imaginaria del paquete de onda en intervalos de  $1 fs$  a lo largo de  $20 fs$ , obtenidas con la Ecuación 3.78. En la Figura 3.10 se muestra la evolución temporal de la densidad del protón.

► Evolución Temporal: Potencial y Densidad

Figura 3.8: Propagación del paquete de onda  $\psi(r, t)$  parte real.Figura 3.9: Propagación del paquete de onda  $\psi(r, t)$  parte imaginaria.

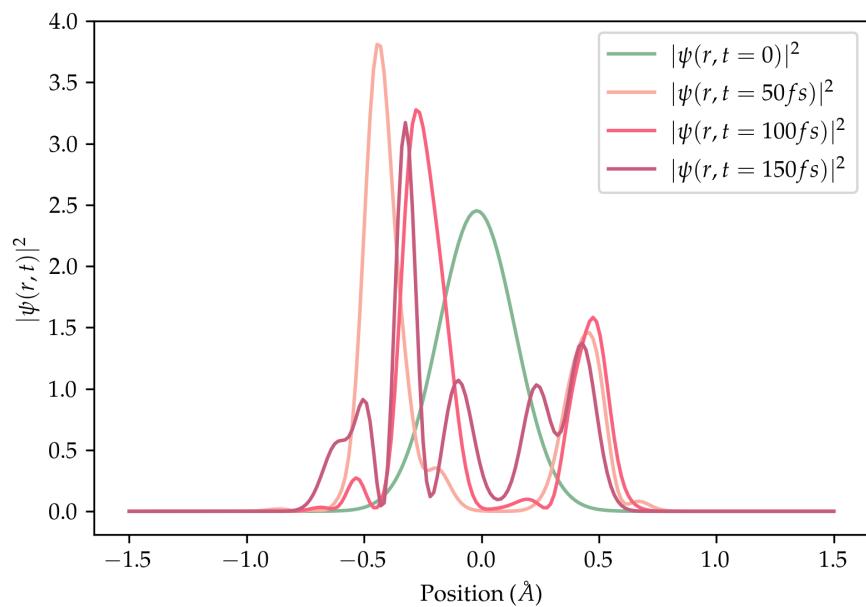


Figura 3.10: Evolución temporal de la densidad  $|\psi(r,t)|^2$ .

## Parte III

### REDES NEURONALES ARTIFICIALES (ANNS)

# 4

---

## APRENDIZAJE AUTOMÁTICO

La **inteligencia artificial**, o AI por sus siglas en inglés, se puede definir como un sistema capaz de interactuar con su entorno. Algunos ejemplos de inteligencias artificiales son: *Siri* de Apple y *Alexa* de Amazon. Para poder generar una respuesta al entorno, estos sistemas contienen sensores que permiten la entrada de información, en estos ejemplos la información es obtenida mediante la voz o las palabras escritas de los usuarios, esta información es procesada a través de métodos de aprendizaje automático o ML por sus siglas en inglés.

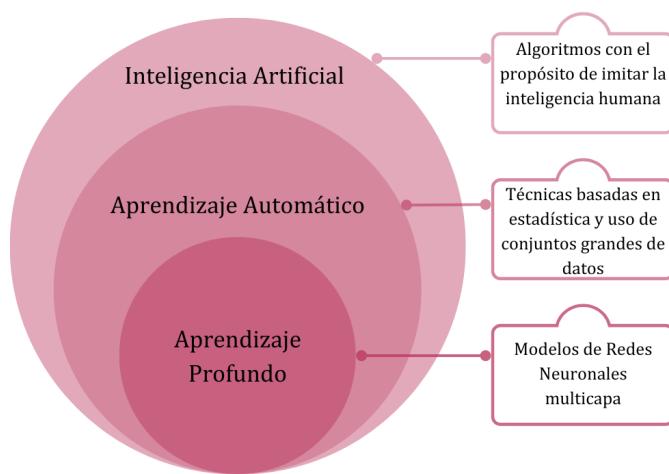


Figura 4.1: Diagrama de Venn que muestra los conceptos de inteligencia artificial, aprendizaje automático y aprendizaje profundo.

Los métodos de **aprendizaje automático** se pueden definir como un conjunto de métodos que pueden detectar automáticamente patrones en datos y aplicarlos para predecir nuevos datos. A grandes rasgos, los métodos de aprendizaje automático pueden dividirse en dos categorías: aprendizaje supervisado y aprendizaje no supervisado. [22]

#### 4.1 APRENDIZAJE SUPERVISADO

En el **aprendizaje supervisado** el objetivo es aprender a partir de un conjunto de  $M$  datos de entrenamiento definidos como:

$$\{\vec{x}_i, y_i\}_{i=1}^M \quad (4.1)$$

una forma de mapear  $\vec{x}_i$  a  $y_i$ .

Cada vector:

$$\vec{x}_i = (x_1, x_2, \dots, x_d)_i \quad (4.2)$$

corresponde a un dato, en donde cada componente es una **característica** o **atributo** del dato en cuestión, y el número de componentes depende del problema. Algunos ejemplos de  $\vec{x}_i$  son:

- Imágenes
- Enunciados de texto
- Audios de voz
- Series de tiempo

Por otro lado,  $y_i$  corresponde a la **etiqueta** de  $\vec{x}_i$ . Cuando  $y_i$  puede tomar un valor categórico, es decir, que de un conjunto finito:

$$y_i \in \{1, \dots, c\}, \text{ por ejemplo: } 1=\text{perro}, 2=\text{gato}, \text{ etc.}$$

se dice que se trata de un problema de **clasificación**. Por otro lado, si  $y_i$ <sup>1</sup> es un valor real, se dice que el problema es de **regresión**.

Algunos ejemplos de métodos de aprendizaje supervisado son:

- Árboles de desición: Para problemas de clasificación
- Regresión lineal: Para problemas de regresión
- Redes Neuronales: Para problemas de regresión y de clasificación

---

<sup>1</sup> $y_i$  puede ser también un vector:  $\vec{y}_i$ , en donde su dimensión está determinada por el problema a resolver, y no necesariamente es igual a la de  $\vec{x}_i$ .

## 4.2 APRENDIZAJE NO SUPERVISADO

En el **aprendizaje no supervisado** el conjunto de entrenamiento de  $M$  datos se reduce a:

$$\{\vec{x}_i\}_{i=1}^M$$

en donde no se cuenta con una etiqueta  $y_i$ . En este tipo de problemas, los métodos están enfocados en buscar patrones importantes a partir de únicamente los datos de entrada.

Algunos ejemplos de métodos de aprendizaje no supervisado son:

- Clustering: Agrupar datos similares entre sí
- Análisis de Componentes Principales: Buscar la relación entre las características de los datos y reducir su dimensionalidad

El **aprendizaje por refuerzo** es comúnmente considerado una tercer categoría de aprendizaje automático, de manera general, en estas técnicas el sistema inteligente interactúa con su entorno con el objetivo de obtener recompensas, bajo el contexto en el que se está implementando, a menudo se utiliza este tipo de aprendizaje para enseñar a las máquinas cómo jugar video juegos. [31]

## 4.3 APRENDIZAJE PROFUNDO

El **aprendizaje profundo**, o **deep learning** en inglés, como se muestra en la [Figura 4.1](#), es un sub-campo del aprendizaje automático, que puede definirse como aquellos métodos que procesan la información en múltiples capas, en donde el nivel de abstracción y complejidad de la información incrementa conforme avanza de una capa a la siguiente.

En la práctica, los modelos de aprendizaje profundo son redes neuronales artificiales multicapa ([Capítulo 5](#)). Algunos ejemplos comúnmente utilizados son: [31]

- Perceptrones Multicapa § 5.2.4
- Redes Neuronales Convolucionales
- Redes Neuronales Recurrentes § 6.2

En el siguiente capítulo se abordarán conceptos básicos acerca de las redes neuronales artificiales, cómo están construidas, es decir, su arquitectura; y cómo, de manera general, este tipo de algoritmos aprenden con base al procesamiento de múltiples datos o muestras de ejemplo.

# 5



## REDES NEURONALES ARTIFICIALES

### 5.1 INTRODUCCIÓN

En el capítulo anterior se abordaron conceptos básicos del aprendizaje automático, así como dos de sus principales enfoques: aprendizaje supervisado y no supervisado. Las redes neuronales artificiales o ANNs por sus siglas en inglés, son comúnmente un método de aprendizaje supervisado<sup>1</sup>, que han tenido gran impulso en los últimos años debido al incremento de datos y a su fácil acceso, así como al crecimiento en poder computacional.

Las **redes neuronales artificiales** son algoritmos de aprendizaje automático que simulan<sup>2</sup> el mecanismo de aprendizaje de los organismos biológicos, en donde las neuronas, es decir, las células del sistema nervioso, se conectan unas con otras mediante las dendritas y los axones en la región espacial nombrada sinapsis Figura 5.1, en donde las conexiones sinápticas a menudo cambian en respuesta a estímulos externos del organismo, proceso que a grandes rasgos es como aprenden los seres vivos. [1]

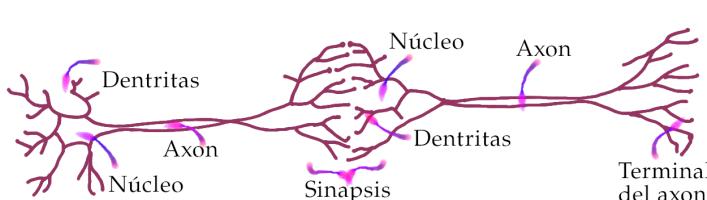


Figura 5.1: Neuronas biológicas transmitiendo información [8]

<sup>1</sup>Las redes neuronales artificiales son algoritmos que también se pueden aplicar en métodos de aprendizaje no supervisado.

<sup>2</sup>Las redes neuronales artificiales a menudo son consideradas más como una caricatura de las biológicas, pues su complejidad rebasa el entendimiento e interpretación que se le puede dar a través de los modelos de ANNs.

## 5.2 ARQUITECTURA BÁSICA

### 5.2.1 El perceptrón

La [Figura 5.2](#) muestra un diagrama de un modelo de perceptrón, que es el modelo más simple de las [ANNs](#). En este ejemplo la **capa de entrada**, o **input layer** en inglés, tiene dos componentes:

$$(x_1, x_2)$$

Es decir,  $d = 2$  en la [Ecuación 4.2](#). La constante 1 es añadida para asignarle el **sesgo** o **bias** en inglés, denotado por:  $b \in \mathbb{R}$ , un parámetro que a menudo se emplea por razones de estadística. La **neurona**, **nodo**, o **node** en inglés, es la unidad computacional que calcula:

$$f\left(\sum_{i=1}^d w_i \cdot x_i + b\right) \quad (5.1)$$

donde los  $w_i \in \mathbb{R}$  son conocidos como los **pesos**, o **weights** en inglés, de  $x_i$ , y  $f$  es una función [§ 5.2.2](#).

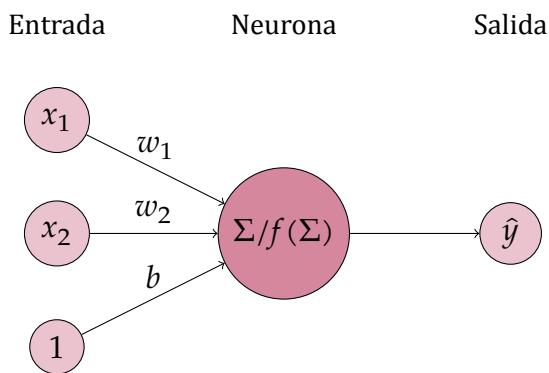


Figura 5.2: Diagrama de un modelo simple de perceptrón

El objetivo del algoritmo, es que mediante el entrenamiento de la red [§ 5.3](#) con un conjunto grande de  $M$  datos:  $\{\vec{x}_j, y_j\}_{j=1}^M$  se obtengan los  $w_i$  óptimos para que se cumpla:

$$y_l = \hat{y}_l = f\left(\sum_{i=1}^d w_i \cdot x_l\right)$$

con  $y_l$  la etiqueta, o valor esperado, correspondiente al vector  $\vec{x}_l$ , en donde  $(\vec{x}_l, y_l)$  es un dato que no necesariamente pertenece al conjunto de entrenamiento que utilizó la red. Esta última propiedad es llamada **generalización del modelo**, y es lo que permite hacer predicciones sobre nuevos datos.

Para simplificar la lectura, en las siguientes secciones y capítulos se utilizará la palabra "red" para hacer referencia a la red neuronal artificial o ANN.

### 5.2.2 Función de activación

En la [Ecuación 5.1](#) la función  $f$  es conocida como una **función de activación** o **activation function** en inglés. Algunos ejemplos de funciones de activación comúnmente utilizadas se muestran en la [Figura 5.3](#).

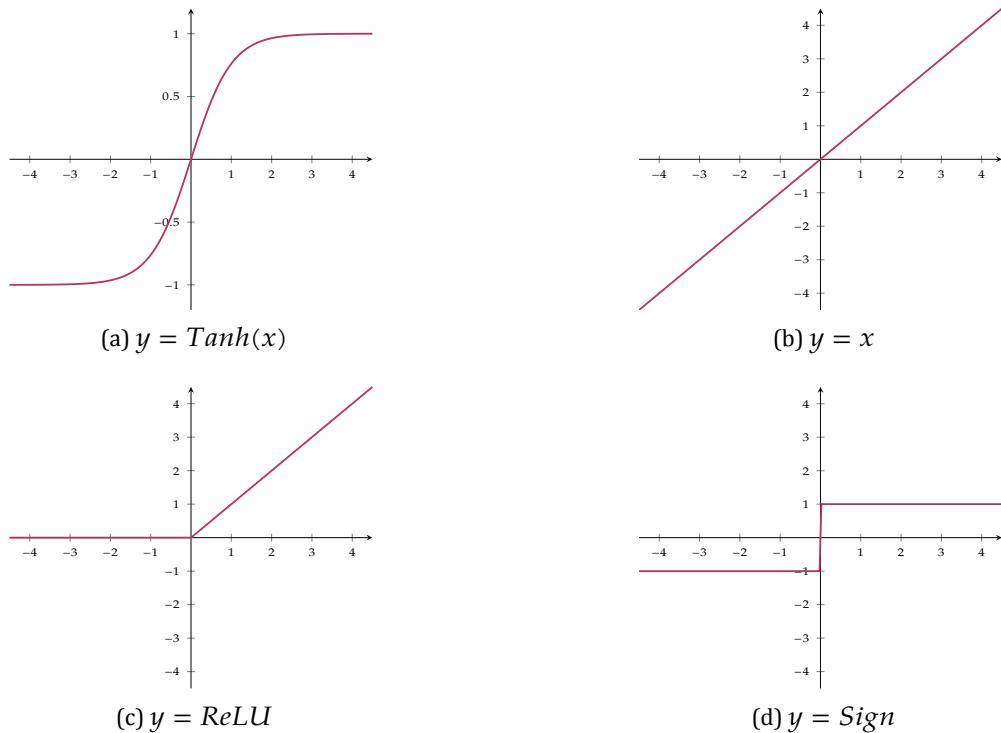


Figura 5.3: Ejemplos de funciones de activación

La importancia en la elección de una función de activación u otra es más notable en los siguientes casos:

1. En la capa de salida de cualquier modelo de red
2. En los modelos multicapa [§ 5.2.4](#)

En el primer caso, la función de activación está motivada por el tipo de valores que puede tomar la etiqueta  $y$ . En problemas de clasificación binaria, las funciones de activación comúnmente utilizadas son *Tanh* o *Sigmoid*, en problemas de clasificación múltiple la función *Softmax*; cuando  $y$  toma valores reales la función de activación utilizada es la *Identidad* o *Lineal* [Figura 5.3b](#).

En el segundo caso, la importancia de elegir una función de activación no lineal entre capas para modelos de clasificación, por ejemplo, permite que el modelo pueda transformar un problema que inicialmente no es linealmente separable, a uno que sí lo es.

### 5.2.3 Función de pérdida

La **función de pérdida**, o **loss function** en inglés:  $L(\hat{y}, y)$  es la función que compara el resultado o salida de la red:  $\hat{y}$  respecto a la etiqueta original  $y$  correspondiente a la entrada  $\vec{x}$ , en otras palabras, indica qué tan bueno es el modelo implementado. El modelo es mejor cuanto menos sea la diferencia entre el valor predicho  $\hat{y}$  respecto al valor esperado  $y$ .

En el entrenamiento de una red, el objetivo es minimizar esta función a través de la actualización de los pesos  $w_i$ . La elección de la función de pérdida, depende del problema a resolver, como en el caso de la función de activación de la última capa. Para modelos en donde los valores de la salida  $\hat{y}$  son reales, se puede utilizar una *función de pérdida cuadrática* o *mean square error (MSE)* en inglés:

$$L = (\hat{y} - y)^2 \quad (5.2)$$

Para problema de clasificación binaria se puede utilizar:

$$L = \log(1 + \exp(-y \cdot \hat{y})) \quad (5.3)$$

mientras que para problemas de múltiples categorías, donde  $\hat{y} = (\hat{y}_1, \dots, \hat{y}_k)$  son las probabilidades<sup>3</sup> de las  $k$  clases<sup>4</sup> una función de pérdida que se puede aplicar es:

$$L = -\log(\hat{y}_r) \quad (5.4)$$

con  $r$  la clase a la que corresponde la etiqueta  $y$ .

### 5.2.4 Modelos multicapa

En la sección § 5.2.1 se revisó el perceptrón como el modelo de red más simple. La Figura 5.4 muestra un diagrama de un modelo multicapa, estos modelos tienen capas ocultas entre las capas de entrada y salida, cada capa contiene un número determinado de neuronas o nodos. El número de capas ocultas y los nodos en cada una, son **hiper-parámetros** de la red, y en general<sup>5</sup>, no se tiene una manera determinista para conocer los valores óptimos en cada modelo.

La arquitectura específica de las redes multicapa se conoce como *redes feed-forward* en inglés, pues las capas se alimentan una tras otra desde la capa de entrada hasta

---

<sup>3</sup>Las probabilidades de cada clase se pueden obtener aplicando la función de activación *Softmax*.

<sup>4</sup>Las **clases** se refieren a las diferentes opciones de salida que puede tener una red de un problema de clasificación múltiple, por ejemplo si la red reconoce tres tipos de fruta, cada fruta es una clase distinta.

<sup>5</sup>*Hyperparameter grid search* es un método que se puede utilizar para conocer los hiperparámetros óptimos de un modelo en particular, sin embargo, este proceso generalmente implica costos computacionales mayores.

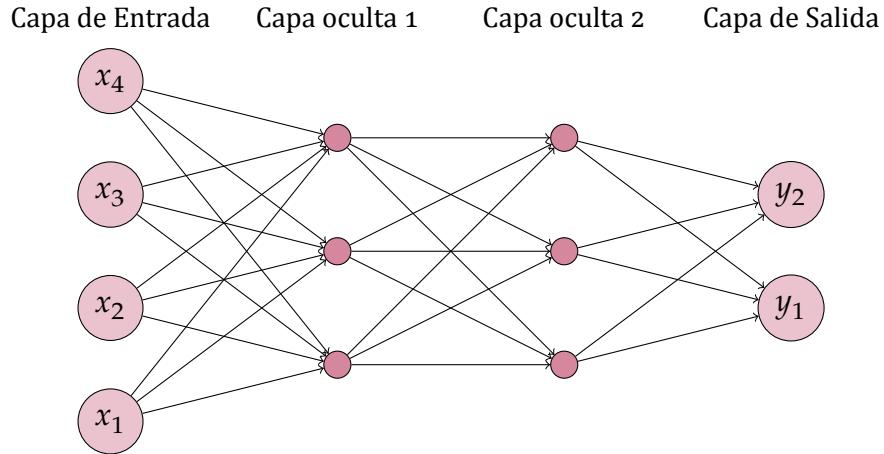


Figura 5.4: Diagrama de un modelo de ANN multicapa con dos capas ocultas.

la capa de salida. La arquitectura por defecto de las redes feed-forward supone que todos los nodos de una capa están conectados a los de la capa siguiente. [1]

La operación que realiza una red multicapa de la capa de entrada a la primer capa oculta con  $p_1$  nodos está definida como:

$$\vec{h}_1 = f(W_1^T \cdot \vec{x})$$

onde  $W_1^T$  es la transpuesta de  $W_1$ , de dimensión  $p_1 \times d$ , y  $\vec{x}$  el vector de entrada de dimensión  $d$ . La operación que se realiza de la capa oculta  $p$  a la  $p + 1 \forall p \in \{1, \dots, k - 1\}$  es:

$$\vec{h}_{p+1} = f(W_p^T \cdot \vec{h}_p)$$

donde  $W_p^T$  es de dimensión  $p_{r+1} \times p_r$  y  $\vec{h}_p$  es de dimensión  $p$ . Finalmente, de la última capa oculta a la capa de salida con dimensión  $o$  la operación realizada es:

$$\vec{o} = f(W_{k+1}^T \cdot \vec{h}_k)$$

con  $W_{k+1}^T$  de dimensión  $o \times p_k$ , y  $\vec{h}_k$  de dimensión  $p_k$ . En todas las expresiones  $f$  es una función de activación que se aplica a cada elemento del vector.

### 5.3 ENTRENAMIENTO DE UNA RED NEURONAL

Una vez construida la arquitectura de una red, el objetivo es obtener los parámetros  $w$  y  $b$  de cada nodo y en cada capa que minimicen la función de pérdida  $L$ . El algoritmo que se utiliza para minimizar la función de pérdida es conocido como **descenso del gradiente estocástico** o SGD por sus siglas en inglés. La Figura 5.5 muestra un sencillo ejemplo del funcionamiento del algoritmo, en donde la función de pérdida únicamente depende de un parámetro  $w$ , el peso inicial es un valor alea-

torio  $w_1$ , luego se calcula el valor del gradiente (en este caso unidimensional: la derivada) y se actualiza el peso como:

$$w_2 = w_1 - \alpha \frac{dL}{dw} \Big|_{w_1}$$

en donde  $\alpha$  es un hiper-parámetro de la red llamado **tasa de aprendizaje o learning rate** en inglés. El proceso continúa actualizando  $w$  hasta llegar a  $w_m$ , el mínimo de la función de pérdida, pues el algoritmo actualiza los pesos en dirección opuesta al gradiente, que apunta a la dirección de máximo cambio.

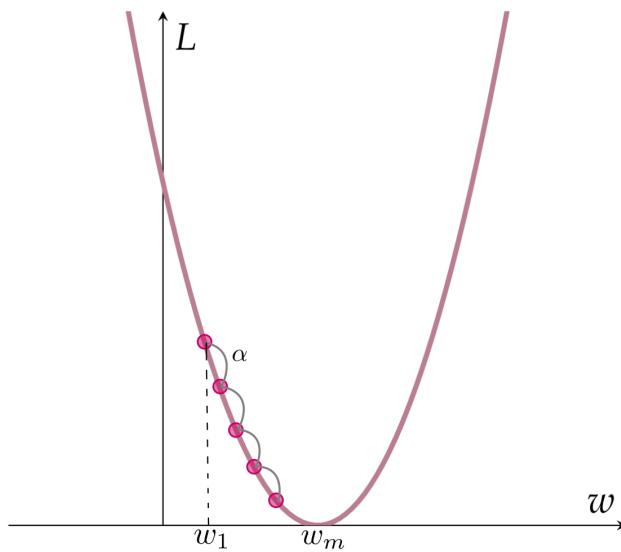


Figura 5.5: Algoritmo de descenso del gradiente para una función de pérdida  $L(w)$

Para modelos de una capa el proceso del descenso del gradiente es directo, pues la función de pérdida depende de los pesos de la única capa, sin embargo, para modelos multicapa la función de pérdida es el resultado de una composición de múltiples funciones de los pesos de las capas anteriores, el algoritmo utilizado para calcular el gradiente en función de la composición se llama **retropropagación o backpropagation** en inglés, que se compone de dos fases principales:

1. Fase de avance: La red recibe las entradas con pesos establecidos de manera aleatoria, se calcula la función de pérdida y las derivadas respecto a la última capa, es decir, la capa de salida.
2. Fase hacia atrás: En esta fase el objetivo es conocer el gradiente de la función de pérdida respecto a los pesos de las capas anteriores utilizando la regla de la cadena.

La [Figura 5.6](#) muestra un ejemplo de la composición de funciones que se realizan en una red neuronal. En este ejemplo existen dos capas antes de la salida  $o$ .

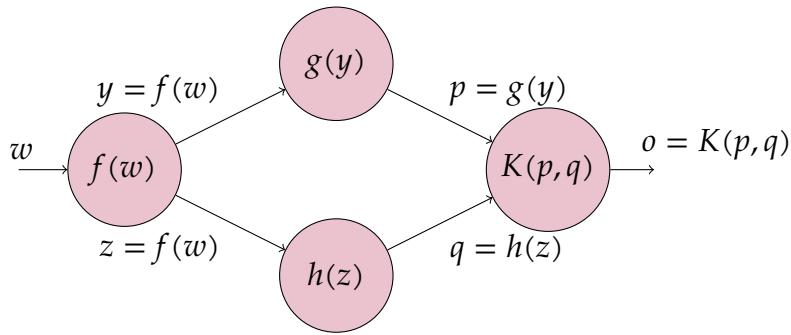


Figura 5.6: Gráfica computacional con dos posibles caminos [1]

Utilizando la regla de la cadena para funciones de varias variables se puede determinar la derivada de  $o$  respecto al peso  $w$ :

$$\frac{\partial o}{\partial w} = \frac{\partial o}{\partial p} \cdot \frac{\partial p}{\partial w} + \frac{\partial o}{\partial q} \cdot \frac{\partial q}{\partial w} \quad (5.5)$$

$$= \frac{\partial o}{\partial p} \cdot \frac{\partial p}{\partial y} \cdot \frac{\partial y}{\partial w} + \frac{\partial o}{\partial q} \cdot \frac{\partial q}{\partial z} \cdot \frac{\partial z}{\partial w} \quad (5.6)$$

$$= \frac{\partial K(p, q)}{\partial p} \cdot g'(y) \cdot f'(w) + \frac{\partial K(p, q)}{\partial q} \cdot h'(z) \cdot f'(w) \quad (5.7)$$

En este ejemplo existen únicamente dos caminos posibles de  $w$  a  $o$ . En la práctica, las redes neuronales multicapa contienen una gran cantidad de caminos posibles entre los pesos de cada capa hasta la salida. Si se considera una secuencia de capas ocultas:  $h_1, \dots, h_k$  seguida de la capa de salida  $o$  respecto a la cual se calcula la función de pérdida  $L$ , la parcial de la función de pérdida respecto al peso  $w_{h_r, h_{r+1}}$  de la conexión de la capa  $h_r$  a la capa  $h_{r+1}$  es: [1]

$$\frac{\partial L}{\partial w_{h_r, h_{r+1}}} = \frac{\partial L}{\partial o} \cdot \left( \sum_{(h_r, h_{r+1}, \dots, h_k, o) \in \mathcal{D}} \frac{\partial o}{\partial h_k} \prod_{i=r}^{k-1} \frac{\partial h_{i+1}}{\partial h_i} \right) \frac{\partial h_r}{\partial w_{h_r, h_{r+1}}} \quad (5.8)$$

con  $\mathcal{D}$  el conjunto de las capas ocultas. Una vez conocidos los gradientes respecto a los pesos en las múltiples capas, se actualizan los valores  $w$  con el objetivo de minimizar la función de pérdida, y así mejorar el modelo.

El número de datos de entrenamiento, es decir de pares:  $(X_l, y_l)$ , que pasan por la red antes de que se actualicen los pesos, se llama **tamaño de lote** o **batch size** en inglés. Una vez que todos los datos de entrenamiento han pasado por la red, se dice que ha pasado una **época** o **epoch** en inglés. Tanto el número de épocas, como el tamaño de lote son hiper-parámetros de la red.

## Parte IV

### IMPLEMENTACIÓN DE UN MODELO DE LSTM COMO PROPAGADOR EN DINÁMICA CUÁNTICA

# 6



## REDES LONG SHORT-TERM MEMORY PARA RESOLVER LA ECUACIÓN DE SCHRÖDINGER

### 6.1 INTRODUCCIÓN

En el capítulo anterior se desarrollaron conceptos básicos generales de las redes neuronales, actualmente existen diversos tipos de redes que se aplican a problemas comunes y que se distinguen por su arquitectura.

En este capítulo se abordará un modelo particular de red llamado *long short-term memory* ([LSTM](#)), que pertenece a un tipo de redes llamadas *redes neuronales recurrentes* ([RNN](#), por sus siglas en inglés). Este tipo de redes es comúnmente utilizada cuando los atributos en el conjunto de datos están relacionados entre sí, o guardan una dependencia.

En la sección § 6.3 se desarrollará la arquitectura de las redes [LSTM](#), así como sus aplicaciones más comunes, haciendo énfasis a las series de tiempo, que son de particular interés en el presente trabajo, pues la evolución temporal de un paquete de onda puede tratarse como un problema de series de tiempo, encontrando así una manera de abordar el problema de propagar una onda en el tiempo utilizando una red [LSTM](#).

En la sección § 6.4 se implementa el modelo de [LSTM](#) que servirá como propagador  $\mathcal{U}$  en la ecuación de Schrödinger Dependiente del Tiempo ([Ecuación 2.7](#)) para los potenciales  $V(r, t)$  revisados en la sección § 3.3.1.

A lo largo de la sección se presentará el conjunto de datos de entrenamiento de la red, así como los parámetros del modelo del potencial que se utilizaron para su obtención; también se presentarán los hiper-parámetros empleados en la red. Finalmente, en la sección § 6.4.4 se presentan los resultados y predicciones obtenidas con del modelo.

## 6.2 REDES NEURONALES RECURRENTES

Como se mencionó anteriormente, las RNNs se emplean cuando entre los atributos del conjunto de datos existen relaciones. Un dato de entrada para este tipo de redes puede verse de la siguiente forma:

$$\vec{X} = (\vec{x}_0, \vec{x}_1, \dots, \vec{x}_n) \quad (6.1)$$

en donde cada  $\vec{x}_t$  es un vector de dimensión  $d$  recibido al tiempo  $t$ . Para cada tiempo  $t$  se tiene un valor de salida:  $\vec{y}_t$  con la misma dimensión.

Algunos ejemplos de aplicaciones se muestran a continuación:

- *Secuencias de texto:* En donde la importancia o información de una palabra depende del contexto en el que se encuentra dentro de una oración. Cada palabra<sup>1</sup>:  $\vec{x}_t$ , puede verse como un atributo relacionado a la palabra anterior en una oración:  $\vec{x}_{t-1}$ , o a la palabra siguiente:  $\vec{x}_{t+1}$ .
- *Series de tiempo:*  $\vec{x}_t$  en este caso puede almacenar  $d$  valores, que pueden ser, por ejemplo, los correspondientes al valor de una función  $f(x, t)$  en una malla de  $d$  puntos en el espacio  $x$  al tiempo  $t$ . En este tipo de configuración la salida  $\vec{y}_t$  podría ser la predicción pronosticada de  $\vec{x}_{t+1}$ .

Un diagrama sencillo de una RNN se muestra en la Figura 6.1, en donde un autobucle actualizará el **estado oculto**  $\vec{h}_t$  a cada tiempo  $t$  después de la entrada  $\vec{x}_t$ :

$$\vec{h}_t = \text{Tanh}(W_{xh}\vec{x}_t + W_{hh}\vec{h}_{t-1}) \quad (6.2)$$

donde *Tanh* es la función de activación, que está siendo aplicada a cada elemento del vector resultante de la multiplicación y suma de las matrices de peso  $W$  por los vectores  $\vec{x}_t$  y  $\vec{h}_{t-1}$  para cada tiempo  $t$ . La salida, por su parte, está dada por:

$$\vec{y}_t = W_{hy}\vec{h}_t \quad (6.3)$$

En la Figura 6.1, a la derecha se muestra la representación de la RNN en capas para cada tiempo, en donde es importante notar que las matrices de peso  $W$  son las mismas en cada tiempo  $t$ , de manera que ecuación de recurrencia Ecuación 6.2 aplica la misma función.

Uno de los principales problemas de las RNNs en la práctica es en el entrenamiento, pues el gradiente tiende a desvanecerse o incrementar excesivamente, debido a la inestabilidad que produce la multiplicación de forma recursiva de las matrices

---

<sup>1</sup>En este caso, la dimensión  $d$  hace referencia al número de palabras que tiene el léxico utilizado.

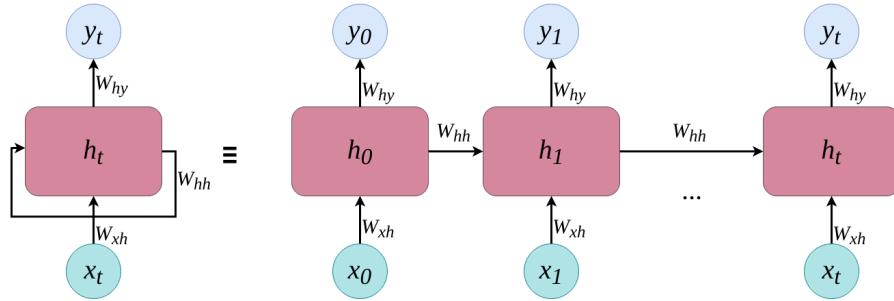


Figura 6.1: Diagrama de una RNN y su representación en capas de tiempo.

de peso  $W$ , problema que puede incrementarse cuanto más grande sea la secuencia de tiempo que procesa la red y cuantas más capas ocultas tenga. [1]

Una alternativa que previene este problema es el cambio de la relación de recurrencia para el estado oculto  $\vec{h}_t$  (Ecuación 6.2) mediante el uso de la *memoria a largo plazo* de una red LSTM, que se abordará en la siguiente sección.

### 6.3 REDES LONG SHORT-TERM MEMORY

Las **redes long short-term memory** son un tipo de RNN diseñadas para mitigar los problemas con el gradiente que estas tienen en el entrenamiento. La manera en la que realizan esto es cambiando las condiciones de recurrencia de cómo el estado oculto  $\vec{h}_t$  es propagado en el tiempo, para ello, se introduce un nuevo vector llamado **celda de estado** o **cell state** en inglés denotado por  $\vec{c}_t$  de la misma dimensión de  $\vec{h}_t$ , que puede interpretarse como una *memoria a largo plazo* que retiene parte de la información de las celdas de estados de tiempos pasados. [1]

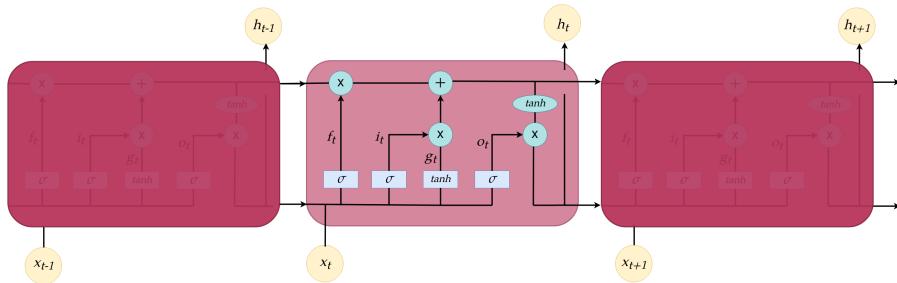


Figura 6.2: Diagrama de una LSTM representada en capas de tiempo.

El diagrama de la Figura 6.2 muestra la estructura de una LSTM desplegada en capas de tiempo, su forma en cadena es igual a la de las RNNs mencionadas en la sección anterior, sin embargo, cada bloque de memoria (recuadros rosas) tiene una arquitectura particular más compleja que la de una RNN común.

### 6.3.1 Arquitectura

La Figura 6.3 muestra un bloque de memoria de una LSTM, en donde se observa cómo interactúan los vectores de estado oculto y de celda de estado pasados:  $\vec{h}_{t-1}$  y  $\vec{c}_{t-1}$  con funciones llamadas compuertas o gates en inglés, denotadas por  $\vec{f}_t$ ,  $\vec{i}_t$ ,  $\vec{o}_t$  y operaciones básicas como suma y multiplicación, para producir nuevos vectores  $\vec{h}_t$  y  $\vec{c}_t$ . Al tiempo  $t = 0$ , los vectores de estado oculto y de celda de estado comienzan inicializados con un valor aleatorio. [24]

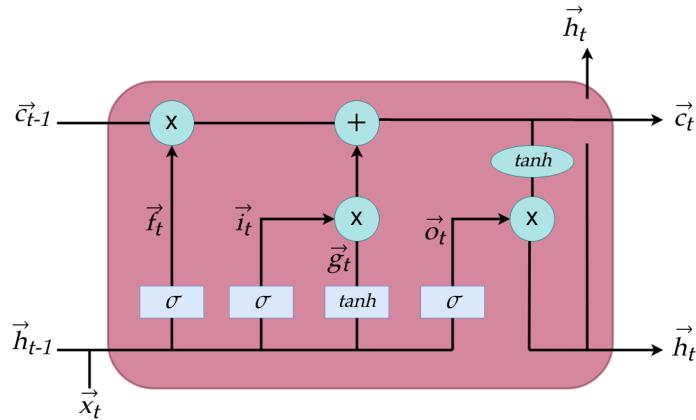


Figura 6.3: Diagrama de un bloque de memoria de una LSTM.

Una compuerta llamada **forget**  $\vec{f}_t$ , decide qué información proveniente del tiempo anterior  $t - 1$  se mantiene y cual no, tomando en consideración también la entrada  $\vec{x}_t$ , aplicando la función sigma (es decir,  $\vec{f}_t$  es una capa *Sigmoid*):

$$\vec{f}_t = \sigma(W_{xf}\vec{x}_t + W_{hf}\vec{h}_{t-1} + b_f) \quad (6.4)$$

En donde  $W$  y  $b$  con sus correspondientes subíndices representan las matrices de peso y vectores de sesgo correspondientes a cada capa oculta.

Posteriormente, para decidir qué nueva información será incluida a la celda de estado se calculan:

$$\vec{i}_t = \sigma(W_{xi}\vec{x}_t + W_{hi}\vec{h}_{t-1} + b_i) \quad (6.5)$$

$$\vec{g}_t = \text{Tanh}(W_{xg}\vec{x}_t + W_{hg}\vec{h}_{t-1} + b_g) \quad (6.6)$$

en donde  $\vec{i}_t$  es una capa *Sigmoid*, llamada compuerta **input**, y  $\vec{g}_t$  es una capa *Tanh* que genera un vector con valores nuevos que son candidatos para ser agregados a la nueva celda de estado  $\vec{c}_t$ .

La actualización de la celda de estado se da como:

$$\vec{c}_t = \vec{f}_t * \vec{c}_{t-1} + \vec{i}_t * \vec{g}_t \quad (6.7)$$

en donde  $*$  representa el *producto Hadamard*, en donde la multiplicación de los vectores se realiza elemento por elemento. Finalmente, la salida esta dada por:

$$\vec{o}_t = \sigma(W_{xo}\vec{x}_t + W_{ho}\vec{h}_{t-1} + b_o) \quad (6.8)$$

$$\vec{h}_t = \vec{o}_t * \text{Tanh}(\vec{c}_t) \quad (6.9)$$

en donde  $\vec{o}$  es una capa *Sigmoid* llamada compuerta **output**, y el vector  $\vec{h}_t$  es el estado oculto al tiempo  $t$ , que servirá como entrada, junto con el vector de celda de estado  $\vec{c}_t$  para el bloque de memoria del tiempo  $t + 1$ , como se muestra en la Figura 6.2.

## 6.4 PROPAGACIÓN TEMPORAL DE FUNCIONES DE ONDA CON LSTM

En esta sección se implementa una red *LSTM* para predecir la evolución en un periodo de tiempo largo<sup>2</sup> de un paquete de onda a un tiempo inicial  $\psi(r, t)$  bajo un potencial dependiente del tiempo  $V(r, t)$  en intervalos de tiempo  $\Delta t$ . Se utilizó como referencia un trabajo previo [29] en donde se implementó un modelo de perceptrón multicapa para obtener el paquete de onda propagado un paso de tiempo  $\Delta t$ .

### 6.4.1 Obtención de datos

El conjunto de datos de entrenamiento utilizado se obtuvo generando 8000 trayectorias de  $200\text{ fs}$ , cada una con un  $\Delta t = 1\text{ fs}$ . El diagrama de la Figura 6.4 muestra una trayectoria, en donde  $\psi(r, t_l)r$  con  $l = 0, \dots, 200$  representa la parte real del paquete de onda,  $\psi(r, t_l)i$  la parte imaginaria y  $V(r, t_l)$  el potencial.

---

<sup>2</sup>Respecto a la escala de tiempo característico del sistema físico estudiado.

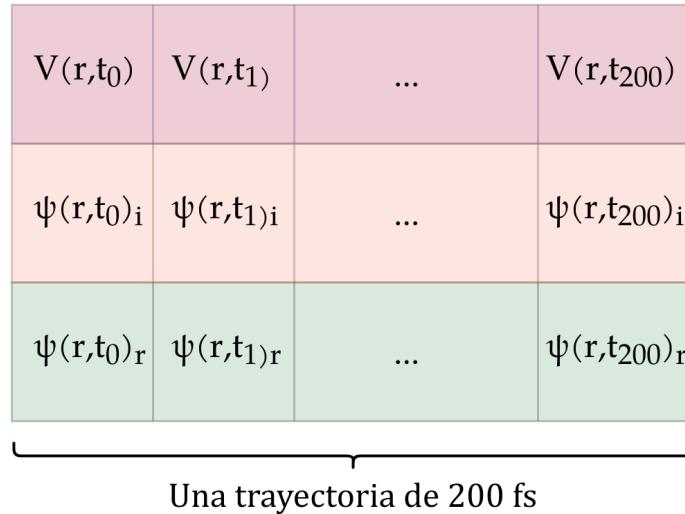


Figura 6.4: Diagrama de una trayectoria generada.

Para cada tiempo  $t_i$  la onda  $\psi(r, t_i)$  contiene el valor de la onda en los  $N = 32$  puntos en el grid del espacio de posiciones en  $r$ .

Para cada trayectoria, los paquetes de onda iniciales, es decir al tiempo  $t = 0$ , se definieron como ondas Gaussianas: (Figura 6.5)

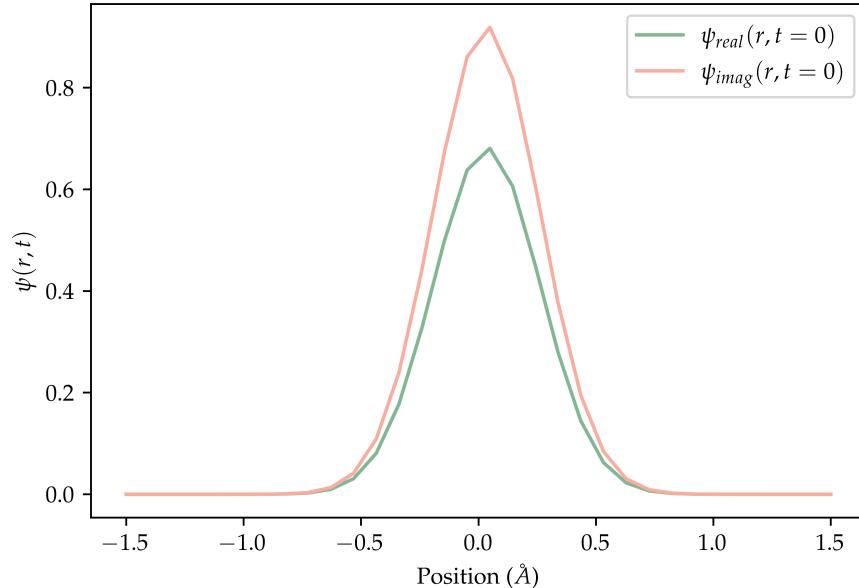


Figura 6.5: Parte real y compleja de un paquete de onda inicial  $\psi(r, t = 0)$ .

$$\psi(r, 0) = C_i \cdot \frac{1}{\sigma \sqrt{2\pi}} \exp\left\{\frac{-(r - \mu)^2}{2\sigma^2}\right\} \quad (6.10)$$

en donde  $\mu$  y  $\sigma$  son valores aleatorios elegidos de una distribución uniforme de  $(-0.5, 0.5)\text{\AA}$  y  $(0.1, 0.3)\text{\AA}$  respectivamente.  $C_i$  es un número complejo aleatorio elegido de tal manera que la onda esté normalizada, es decir:

$$\langle \psi(r, 0) | \psi(r, 0) \rangle = 1$$

Para propagar la onda se utilizó el método DVR revisado en la sección § 3.3 con un grid de  $N = 32$  puntos, con  $a = r_0 = -1.5\text{\AA}$  y  $b = r_{N-1} = 1.5\text{\AA}$  (Ecación 3.54).

El modelo de potencial  $V(r, t)$  utilizado fue el mismo que se usó en la sección § 3.3.1. Para cada trayectoria se eligieron valores de parámetros para el modelo de potencial aleatorios entre rangos especificados en la Tabla A.2. En la Figura 6.6 se muestra un ejemplo de un potencial generado.

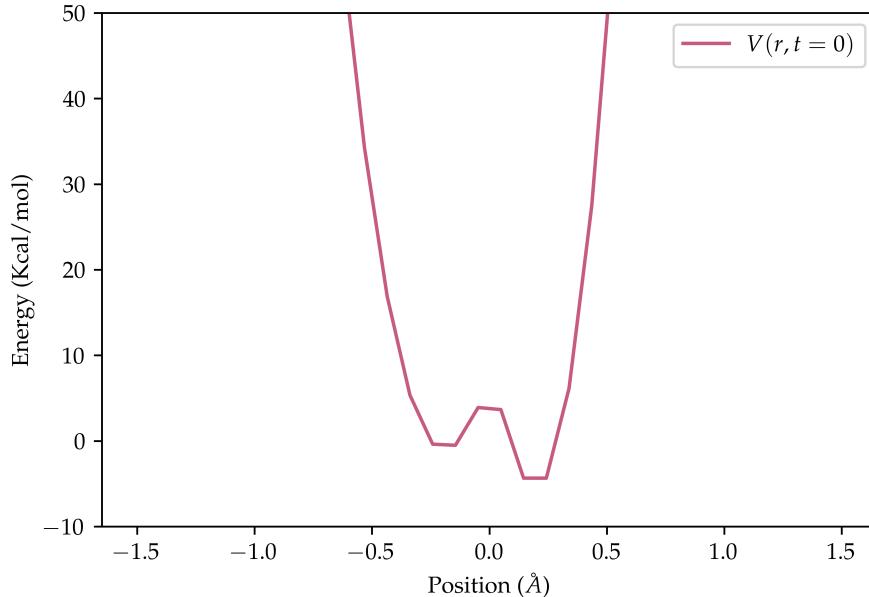


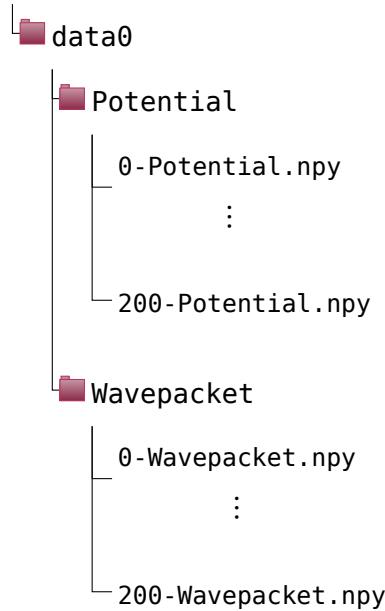
Figura 6.6: Potencial inicial al tiempo  $t = 0$

En el repositorio [Trayectorias](#) se encuentran los datos generados<sup>3</sup>. Cada carpeta contiene los directorios y archivos siguientes:

---

<sup>3</sup>⚠️ Para trabajos futuros, se recomienda fuertemente NO crear un directorio por dato, pues esto generó problemas de peso y tiempo de cómputo. Un solo archivo con extensión .csv o .hdf5 es más recomendado para guardar una mayor cantidad de datos.

## Trayectorias



En donde `data0` corresponde a la trayectoria 0. El archivo `o-Potential.npy` es un arreglo de  $N = 32$  entradas que corresponde al valor del potencial al tiempo  $t = 0fs$ , el archivo `o-Wavepacket.npy` es un arreglo de  $N = 32$  entradas complejas que corresponden al paquete de onda al tiempo  $t = 0fs$ ; así respectivamente para cada tiempo hasta  $t = 200fs$ . Además de los datos, se encuentra un archivo de texto: `ValuesPotential.txt`, en donde se registran los valores de los parámetros para generar el potencial, es decir, los valores exactos elegidos de la [Tabla A.2](#).

El código para generar las trayectorias se encuentra en el repositorio: [Geneación de datos](#), que contiene dos clases: `Potential_System`, que contiene las ecuaciones del modelo del potencial y la clase `ProtonTransfer`, con la que se generan las trayectorias. A continuación se muestra un ejemplo para generar una trayectoria, con la información que muestra el diagrama de la [Figura 6.4](#):

```

trayectoria0 = ProtonTransfer(n=32, a=-1.5, b=1.5, time=True,
    var_random=True, save_dir='data0')
trayectoria0.vector_potential(t=200, step=1)
trayectoria0.evolution_wp(t=200, step=1, gaussiana=True)

```

En donde se utilizan las siguientes variables:

- n: Número de puntos en el grid
- a: Punto inicial del grid [ $\text{\AA}$ ]
- b: Punto final del grid [ $\text{\AA}$ ]

- time: True o False. Determina si se utiliza un potencial dependiente del tiempo: True, o independiente del tiempo: False
- var\_random: True o False. True inicia las variables de manera aleatoria para el potencial del sistema. False solicita al usuario cada variable. [Tabla A.2](#)
- save\_dir: Nombre del directorio donde se guardarán los datos del potencial y la evolución de onda.
- t: Tiempo total de la trayectoria [fs]
- step:  $\Delta t$  para la evolución temporal de la onda [fs]
- gaussiana: True o False. True genera una onda inicial Gaussiana ([Ecuación 6.10](#)), False solicita  $k$  para generar la onda inicial como una suma de eigenfunciones ([Ecuación 3.76](#))

#### 6.4.2 Procesamiento de datos: visualización y forma

Para el entrenamiento de la red, las trayectorias deben estar conformados por entradas  $X$  y etiquetas o salidas  $y$ , para este problema, cada entrada está conformada por la parte real y compleja de un paquete onda y el potencial al tiempo  $t$ , es decir un vector de la forma:

$$(\psi_{real}(r, t), \psi_{imag}(r, t), V(r, t))$$

para  $t \in \{0, 1, \dots, 199\}$  de dimensión:

$$(32 + 32 + 32, 200) = (96, 200)$$

mientras que la salida es el paquete de onda propagado al tiempo  $t + 1$ :

$$(\psi_{real}(r, t + 1), \psi_{imag}(r, t + 1))$$

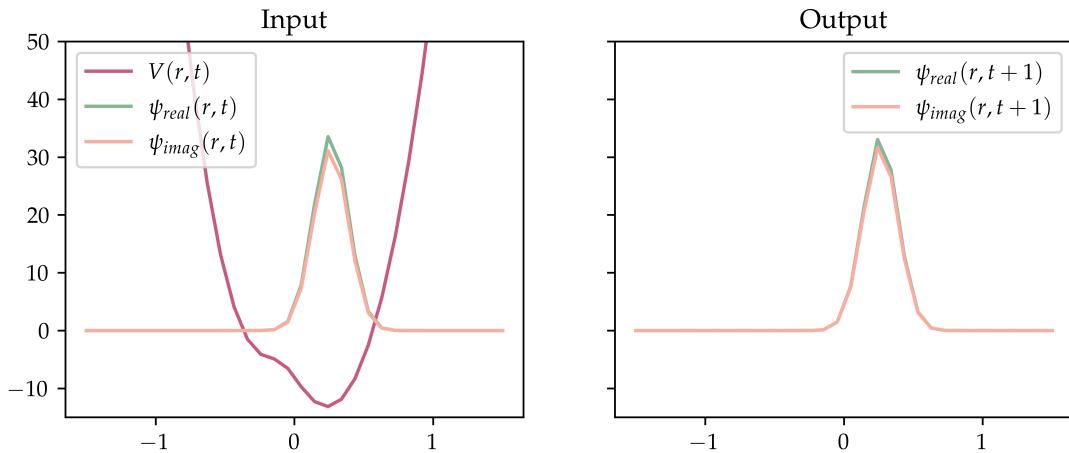
para  $t \in \{0, 1, \dots, 199\}$  de dimensión:

$$(32 + 32, 200) = (64, 200)$$

La [Figura 6.7](#) muestra cada componente de los vectores de entrada y salida en su representación del espacio de posiciones  $r$  a un tiempo  $t$ , en donde las funciones de onda  $\psi$  están escaladas por un factor de 20 para poder visualizar el potencial en la misma gráfica.

El código para la preparación de los datos se encuentra en la [Notebook principal](#) en la clase `Propagator_Dataset`.

La división de los datos se muestra en la [Tabla 6.1](#).

Figura 6.7: Visualización gráfica: Entrada  $X$  y etiqueta  $y$  a un tiempo  $t$ .

ENTRENAMIENTO	VALIDACIÓN	TEST	TOTAL
5600	1600	800	8000
70%	20%	10%	100%

Tabla 6.1: División de datos

#### 6.4.3 Modelo LSTM: arquitectura y entrenamiento

Para el modelo de la red y su entrenamiento se utilizó PyTorch versión 1.9.0+cu102 en una computadora personal. El resumen del modelo se muestra en la Tabla 6.2, y el código completo se puede ver en [Implementación LSTM](#).

En donde los parámetros utilizados fueron elegidos después de varios intentos para hallar los que minimizaran la función de pérdida o loss function más rápido, es decir en un menor número de épocas. Las funciones de activación en las capas LSTM son las mencionadas en la sección § 6.3, mientras que la última capa corresponde a una capa lineal debido a que la salida  $y$  corresponde a valores reales. El algoritmo de optimización<sup>4</sup> utiliza una técnica de regularización para prevenir el sobre-entrenamiento, que consiste en agregar una pequeña penalización a la pérdida, que es la norma  $L_2$  de todos los pesos entrenables del modelo: [21]

$$\text{loss} = \text{loss} + \text{weight decay} * L_2$$

En la Figura 6.8 se muestra un diagrama del modelo de red representado en capas de tiempo (sección § 6.3), con las entradas y salidas correspondientes.

<sup>4</sup>Detalles en la sección § 5.3

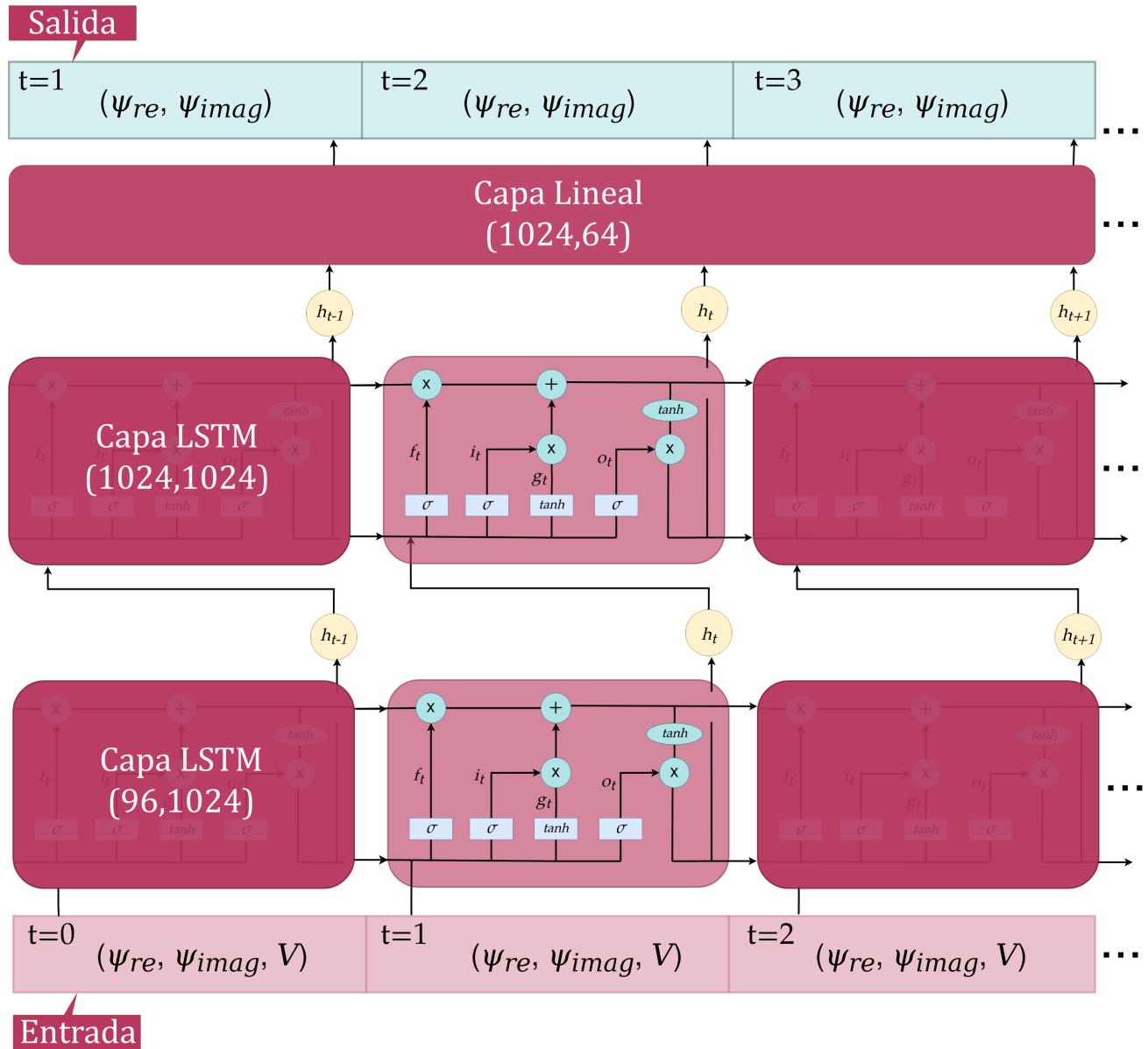


Figura 6.8: Diagrama del modelo LSTM: se muestran 3 de las 200 capas de tiempo.

CAPAS	NODOS	ENTRADA / SALIDA
LSTM	1024	(In: 96, Ou:1024)
LSTM	1024	(In: 1024, Ou:1024)
Lineal	64	(In: 1024, Ou:64)
Loss Function:	Mean Square Error	
Optimizer:	AdamW	weight decay = 0.01
Batch size:	10	
Learning rate:	0.0001	
Epochs:	290	

Tabla 6.2: Resumen del modelo

Para calcular la precisión del modelo se utilizó la [Ecuación 6.11](#):

$$S = \langle \psi_{LSTM} | \psi_{True} \rangle = |S| \exp\{i\theta\} \quad (6.11)$$

que compara dos funciones de onda, la predicha por el modelo  $\psi_{LSTM}$  respecto a la esperada  $\psi_{True}$ , cuando se trata de la misma función de onda, dado que están normalizadas, se sigue que:

$$\langle \psi | \psi \rangle = \int_a^b \psi(x) \psi(x)^\dagger dx = \int_a^b |\psi(x)| dx = 1 = |S| \exp\{i\theta\}$$

$\Leftrightarrow |S| = 1$  y  $\theta = 0$ . Es decir, que la función de onda  $\psi_{LSTM}$  predicha por el modelo es más parecida a la esperada  $\psi_{True}$  cuanto más se acerca  $|S|$  a 1 y  $\theta$  a 0.

Al finalizar cada época en el entrenamiento se calculó la función de precisión para cada onda en cada trayectoria en el conjunto de validación y se promediaron los valores correspondientes a  $|S|$  y  $\theta$ .

#### 6.4.4 Resultados y análisis: precisión y predicciones

En la [Figura 6.9](#) se muestran los valores promedio obtenidos de la magnitud absoluta  $|S|$  y la fase  $\theta$  a través de las épocas.

Al finalizar el entrenamiento, sobre el conjunto de testeo los resultados de las variables se reportan en la [Tabla 6.3](#).

A continuación se muestran algunos ejemplos de predicciones de un solo paso del tiempo tomados aleatoriamente de distintas trayectorias. Los potenciales

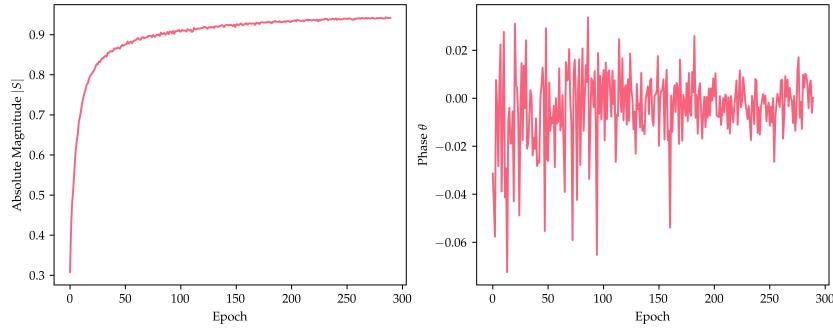


Figura 6.9: Valores promedio de  $|S|$  y  $\theta$  en el conjunto de validación por cada época.

$ S $	$\theta$
0.946	0.0001

Tabla 6.3: Valores promedio de  $|S|$  y  $\theta$  en el conjunto de testeo

$V(r, t)$  y funciones de onda iniciales  $\psi(r, t = 0)$  se tomaron del conjunto de testeo.

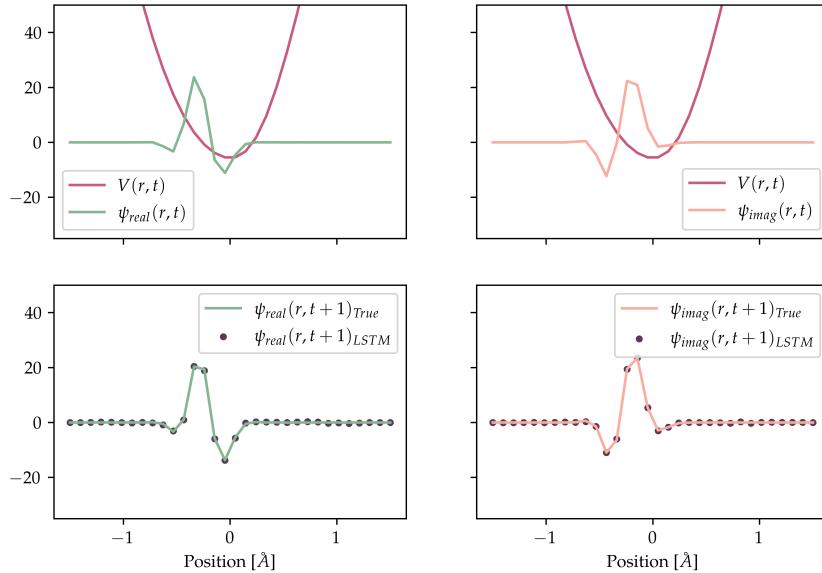


Figura 6.10: Ejemplo 1 de predicción a un paso de tiempo  $\Delta t = 1fs$ .

Las funciones de onda  $\psi$  están escaladas para poder visualizar el potencial en la misma gráfica.

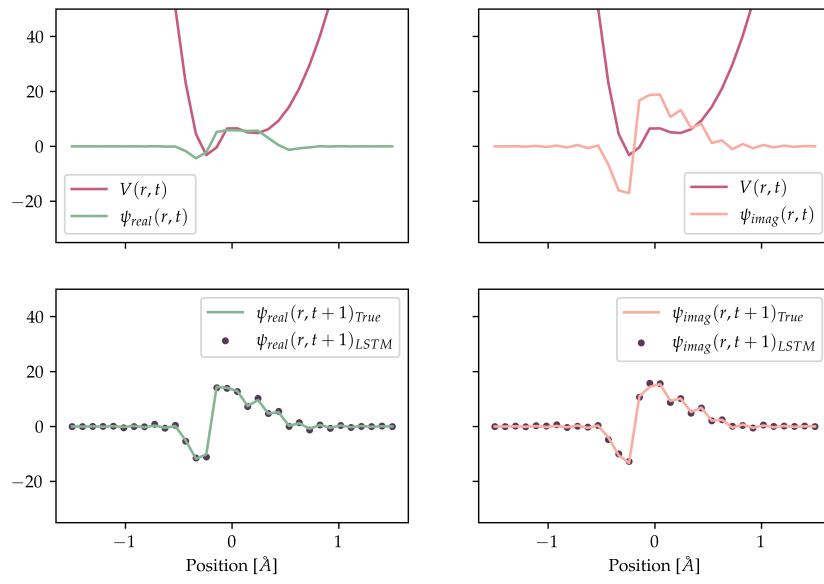


Figura 6.11: Ejemplo 2 de predicción a un paso de tiempo  $\Delta t = 1fs$ .

Las funciones de onda  $\psi$  están escaladas para poder visualizar el potencial en la misma gráfica.

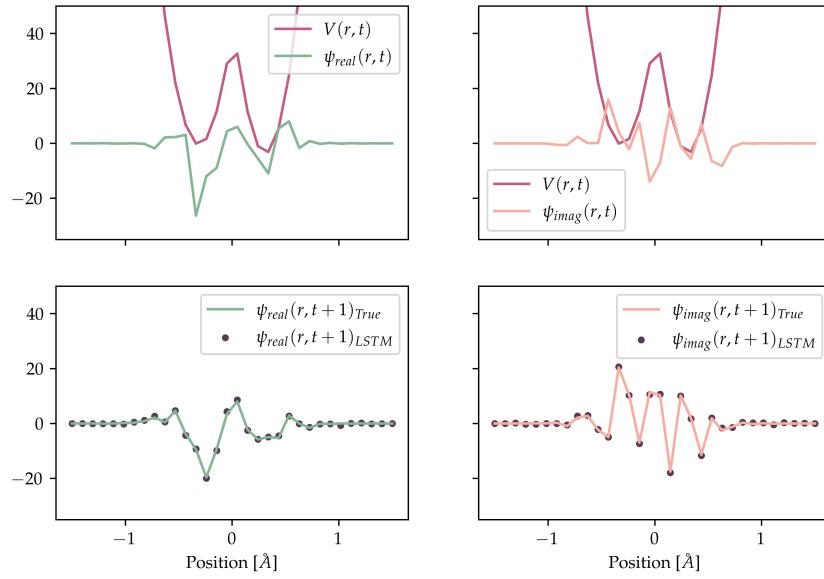


Figura 6.12: Ejemplo 3 de predicción a un paso de tiempo  $\Delta t = 1fs$ .

Las funciones de onda  $\psi$  están escaladas para poder visualizar el potencial en la misma gráfica.

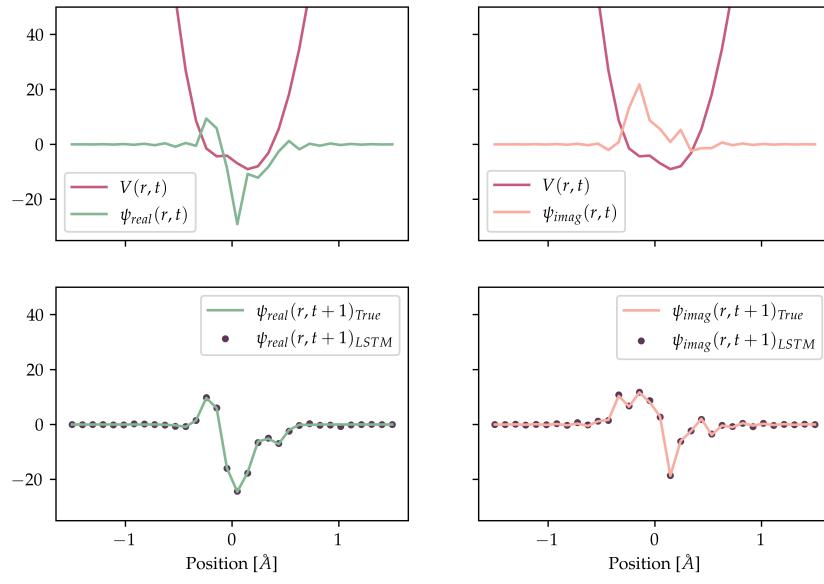


Figura 6.13: Ejemplo 4 de predicción a un paso de tiempo  $\Delta t = 1fs$ .

Las funciones de onda  $\psi$  están escaladas para poder visualizar el potencial en la misma gráfica.

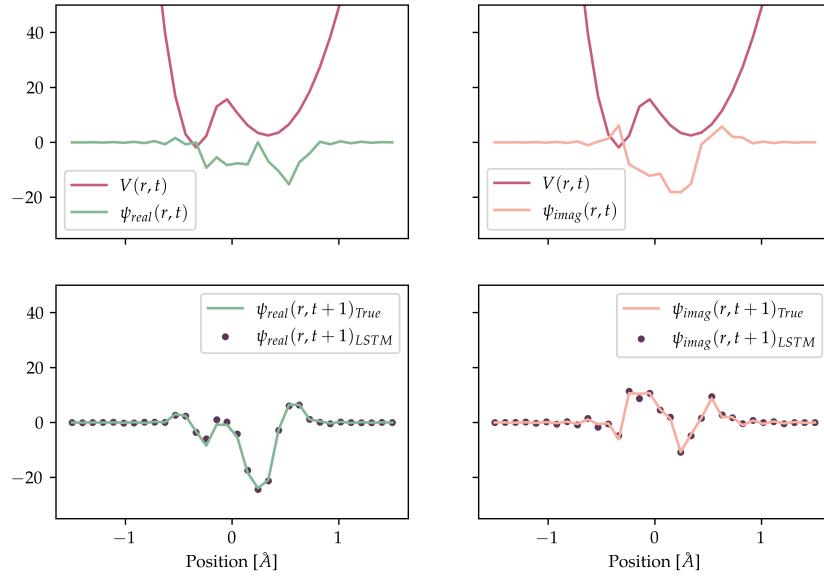


Figura 6.14: Ejemplo 5 de predicción a un paso de tiempo  $\Delta t = 1fs$ .

Las funciones de onda  $\psi$  están escaladas para poder visualizar el potencial en la misma gráfica.

En la [Tabla 6.4](#) se reportan los tiempos de computación requeridos para propagar un paso de tiempo una onda usando los métodos numéricos que utiliza el método [DVR](#): la exponenciación o la diagonalización [29], y el tiempo requerido que utiliza la red [LSTM](#) entrenada.

N GRID	EXPONENCIAL <sup>a</sup>	DIAGONALIZACIÓN <sup>b</sup>	LSTM <sup>c</sup>
32	0.41 s	0.14 s	0.0536 s

<sup>a</sup>Tiempo necesario para realizar la exponenciación matricial del hamiltoniano y multiplicación de matrices para la propagación de paquetes de ondas. <sup>b</sup>Tiempo requerido para realizar la diagonalización de matrices, multiplicación de vectores por elementos, y transformación de base. <sup>c</sup>Tiempo necesario para aplicar la red LSTM entrenada para la propagación de paquetes de ondas.

Tabla 6.4: Tiempo requerido para propagar un paquete de onda un paso en el tiempo  $\Delta t$

Las siguientes gráficas muestran la evolución temporal de la densidad de probabilidad  $|\psi(r, t)|$  a través del tiempo bajo potenciales  $V(r, t)$  calculadas utilizando la función de onda predicha por la red. Los potenciales  $V(r, t)$  y funciones de onda iniciales  $\psi(r, t = 0)$  se tomaron del conjunto de testeo.

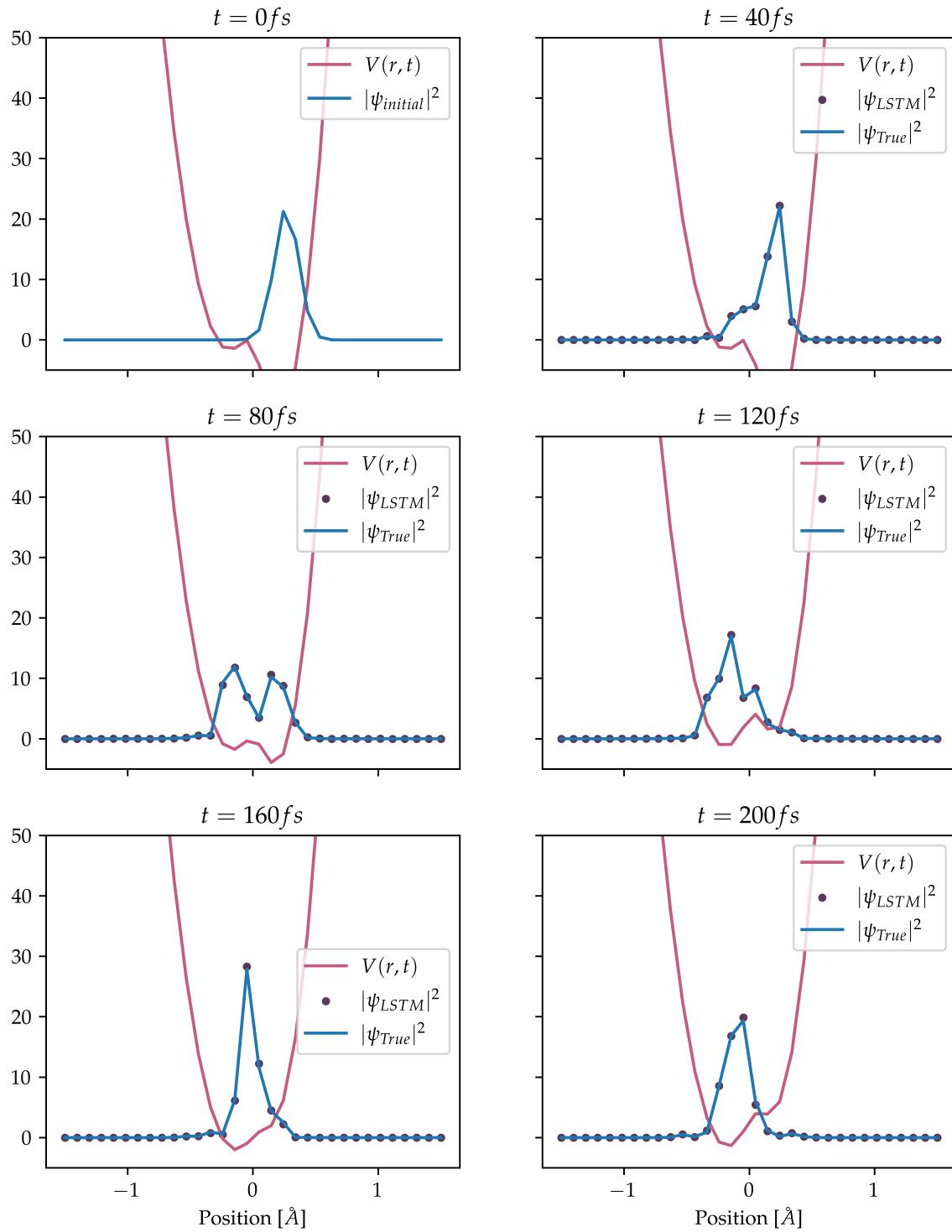


Figura 6.15: Ejemplo 1 de predicción para una trayectoria completa de  $200 fs$ .

Las densidades de probabilidad  $|\psi|$  están escaladas para poder visualizar el potencial en la misma gráfica.

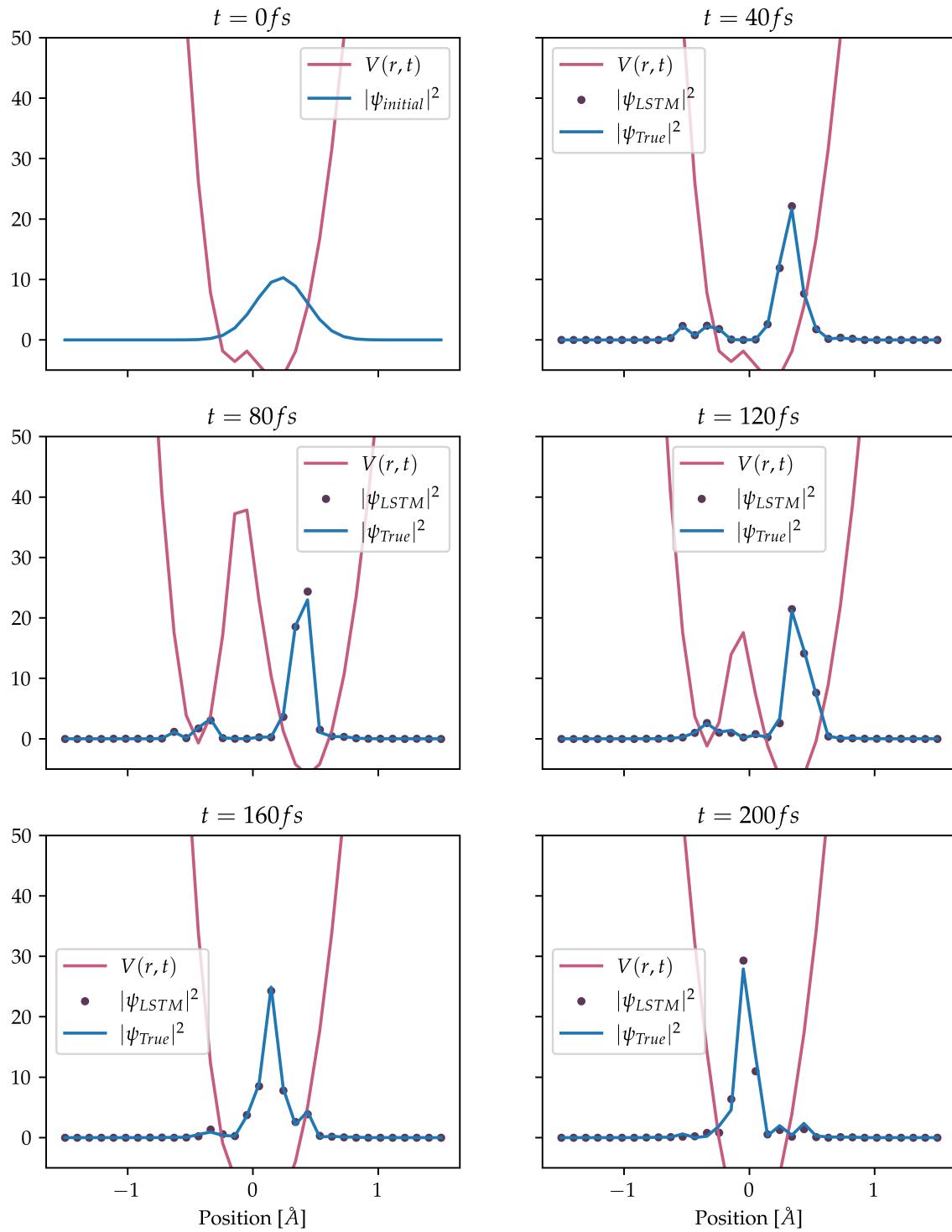


Figura 6.16: Ejemplo 2 de predicción para una trayectoria completa de  $200 fs$ .

Las densidades de probabilidad  $|\psi|$  están escaladas para poder visualizar el potencial en la misma gráfica

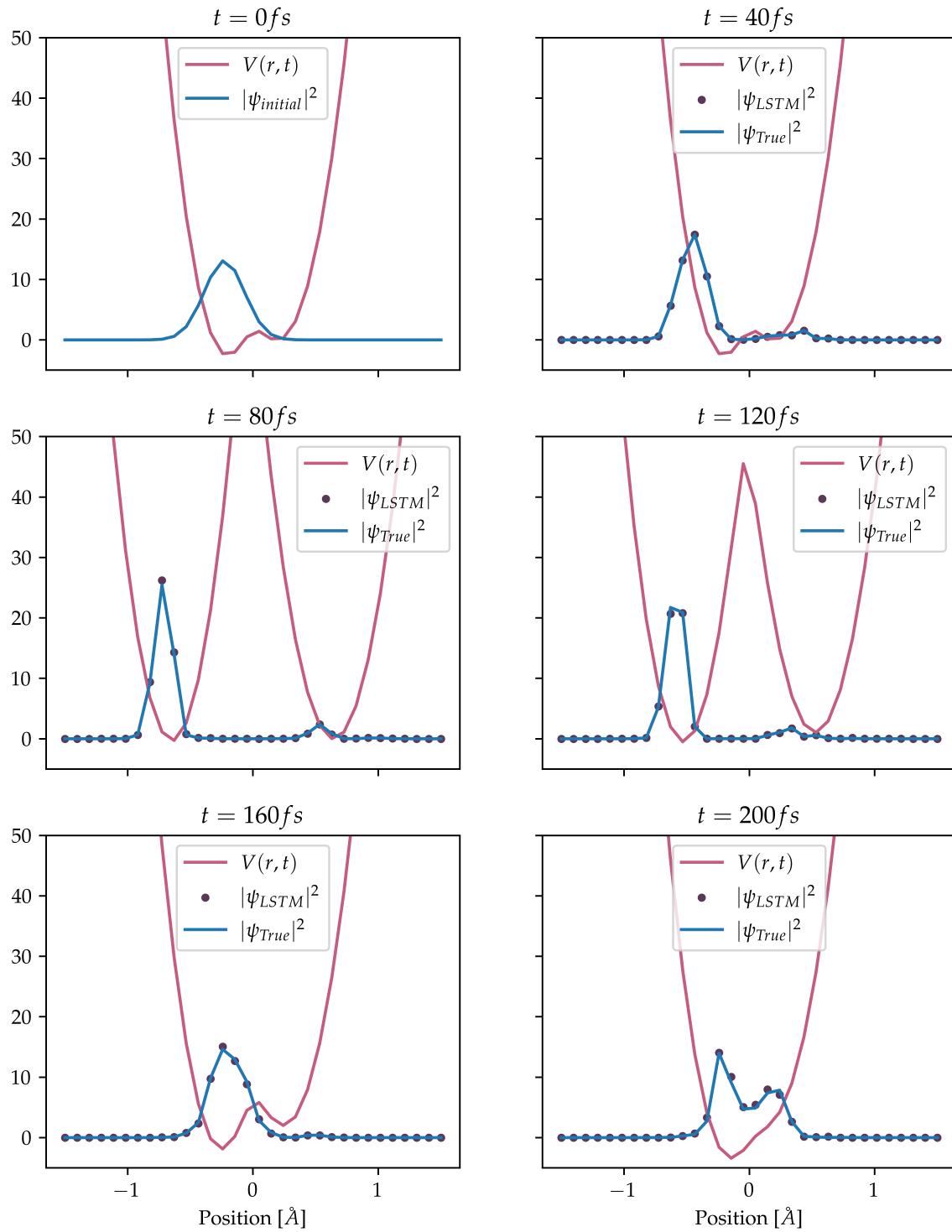


Figura 6.17: Ejemplo 3 de predicción para una trayectoria completa de  $200 fs$ .

Las densidades de probabilidad  $|\psi|$  están escaladas para poder visualizar el potencial en la misma gráfica

En las gráficas anteriores se observa que la red es capaz de predecir la evolución temporal de una onda inicial hasta  $200\text{fs}$  después, dado que la escala de tiempo característico del movimiento vibratorio de protones es de  $10\text{fs}$  a  $25\text{fs}$  [29], la red puede predecir al menos 8 veces más esta escala de tiempo característico.

#### 6.4.5 Aumento de resolución con dos LSTM

Una manera de aumentar la resolución de la red, es decir, aumentar el número de puntos en el grid es tomar una división de  $N = 64$  puntos en el grid para la onda inicial  $\psi(r, t = 0)$  y los potenciales  $V(r, t)$ , y dividirlos en dos grupos de  $N_{azul} = 32$  y  $N_{rojo} = 32$  de manera distribuida en el grid, como se muestra en la Figura 6.18.

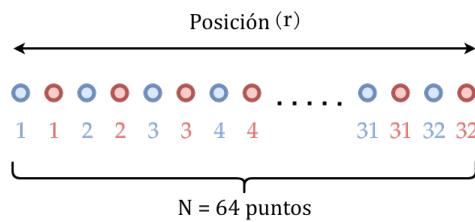


Figura 6.18: División de 64 puntos en el espacio de posiciones en dos grupos.

De esta forma, se tienen los valores correspondientes:  $\psi_{azul}(r, t = 0)$ ,  $V_{azul}(r, t)$ ,  $\psi_{rojo}(r, t = 0)$  y  $V_{rojo}(r, t)$ , en donde cada uno tiene entradas de  $N = 32$  puntos, que se ajustan al tamaño de la entrada de la red. Así, utilizando dos veces la misma red, una vez para cada grupo de puntos, se obtienen los resultados a cada tiempo  $t$  para un grid de tamaño de  $N = 64$  puntos.

Las siguientes gráficas muestran el resultado de aplicar este método.

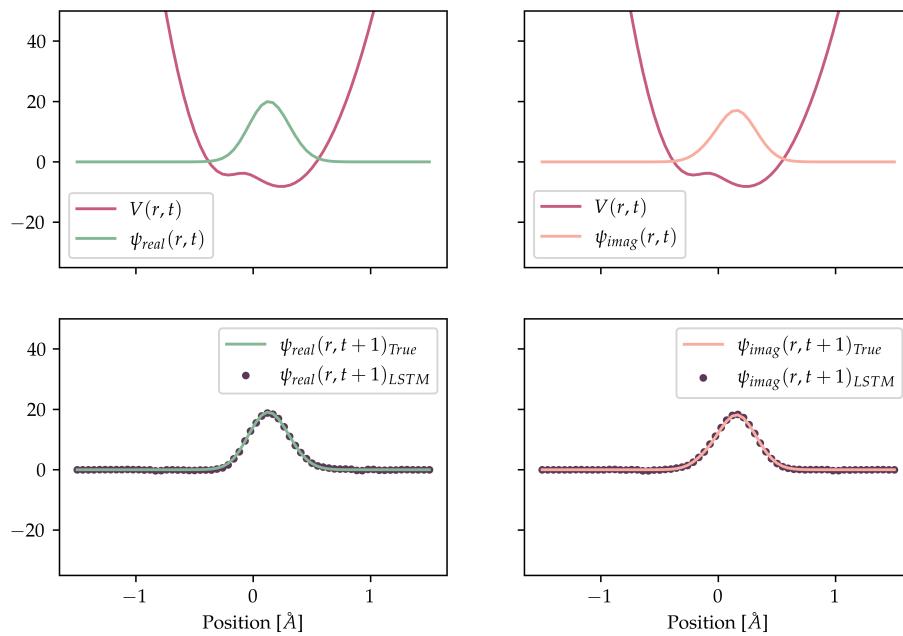


Figura 6.19: Ejemplo 1 de predicción a un paso de tiempo  $\Delta t = 1 fs$ ,  $N = 64$  puntos.

Las funciones de onda  $\psi$  están escaladas para poder visualizar el potencial en la misma gráfica.

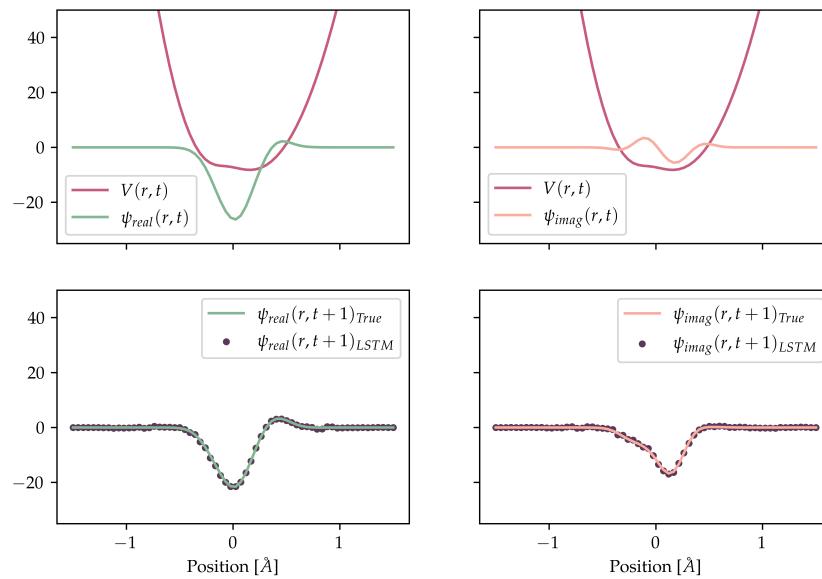


Figura 6.20: Ejemplo 2 de predicción a un paso de tiempo  $\Delta t = 1 fs$ ,  $N = 64$  puntos.

Las funciones de onda  $\psi$  están escaladas para poder visualizar el potencial en la misma gráfica.

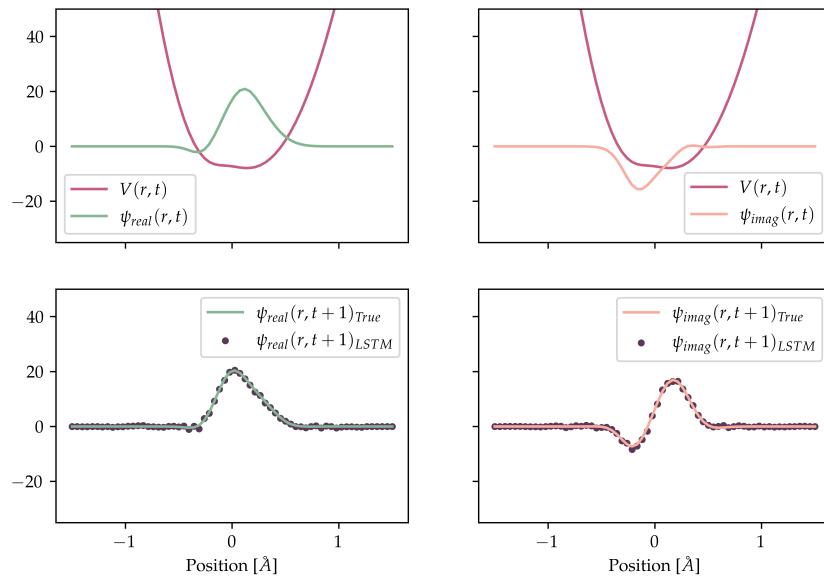


Figura 6.21: Ejemplo 3 de predicción a un paso de tiempo  $\Delta t = 1 fs$ ,  $N = 64$  puntos.

Las funciones de onda  $\psi$  están escaladas para poder visualizar el potencial en la misma gráfica.

La siguiente gráfica muestra la evolución temporal de la densidad de probabilidad  $|\psi(r, t)|$  bajo un potencial  $V(r, t)$  aplicando el método de aumento de resolución de puntos.

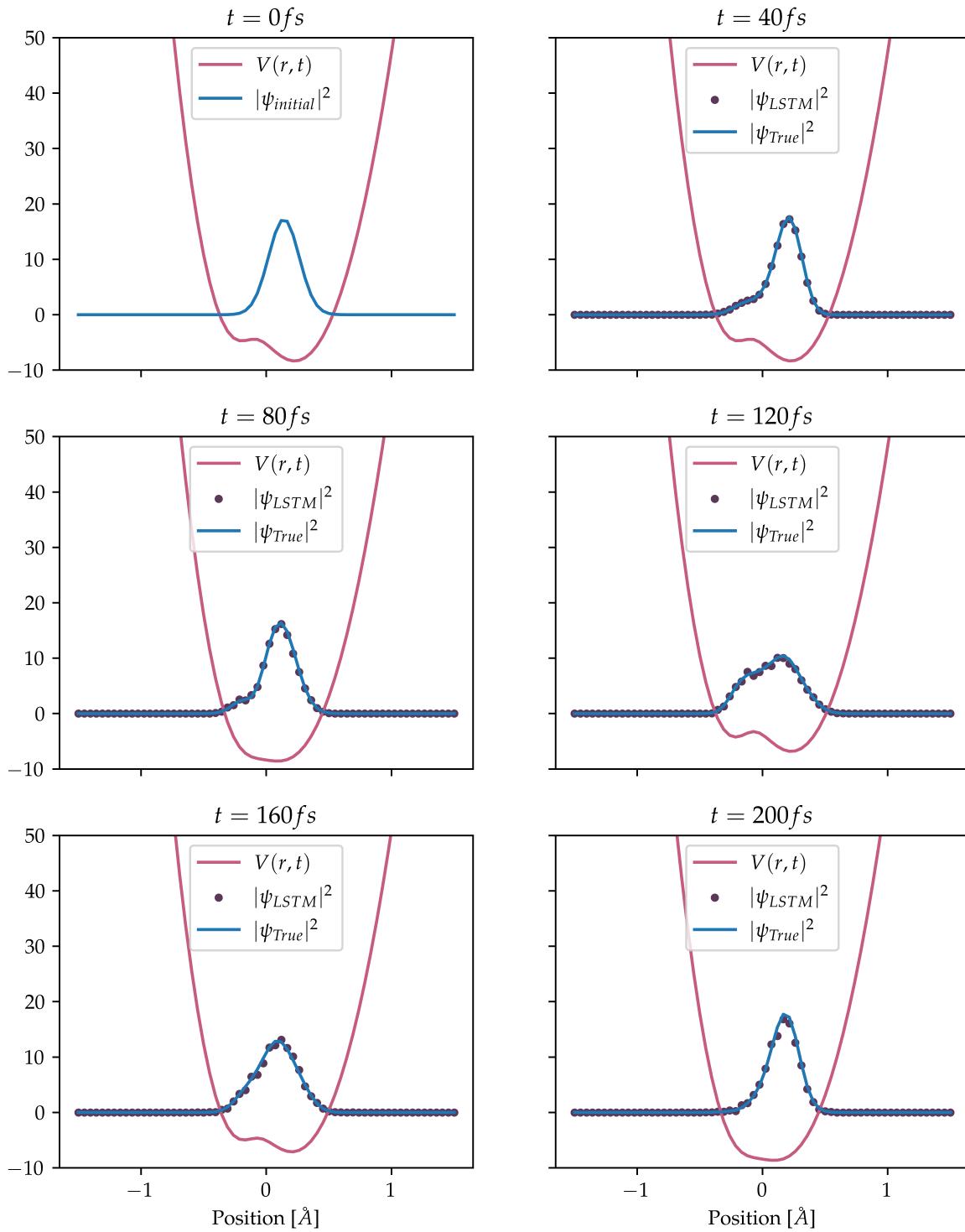


Figura 6.22: Ejemplo de predicción para una trayectoria completa de  $200\text{ fs}$ ,  $N = 64$  puntos.

Las densidades de probabilidad  $|\psi|$  están escaladas para poder visualizar el potencial en la misma gráfica

#### 6.4.6 *Conclusión*

La evolución temporal de un sistema cuántico se encuentra resolviendo la ecuación de Schrödinger [Ecuación 2.7](#), cuando el Hamiltoniano del sistema es dependiente del tiempo, encontrar el operador de propagación, que es una solución general de la ecuación, puede ser un proceso complicado para los métodos analíticos, o costoso en tiempo computacional para los métodos numéricos. En este trabajo se desarrolló una alternativa para resolver la [TDSE](#) utilizando una red neuronal artificial tipo long short-term memory como propagador, encontrando que la red entrenada necesita menos tiempo de ejecución para propagar una onda en comparación a otros métodos numéricos, con una calidad de predicción promedio de 0.946 para la magnitud  $|S|$  y 0.0001 para la fase absoluta  $\theta$ .

Como propuesta para trabajos futuros se presentan las siguientes sugerencias:

- Utilizar más puntos en el grid: Generar datos con más puntos en el grid requieren de un mayor tiempo de computación, sin embargo la calidad en las predicciones puede mejorar significativamente, esperando resultados como los que se presentan en la sección [§ 3.3.1](#).
- Implementar el modelo para dos dimensiones en el espacio de posiciones: En el documento de información adicional de la referencia [\[29\]](#) se proporcionan las ecuaciones para modelar el potencial en dos dimensiones.

## Parte V

### APÉNDICE

# A



## APÉNDICE

### A.1 PRUEBA DE LA IDENTIDAD

$$\sum_1^n \cos nx = \cos \frac{(n+1)x}{2} \frac{\sin \frac{nx}{2}}{\sin \frac{x}{2}}$$

### A.2 VARIABLES DE PARÁMETROS PARA EL POTENCIAL

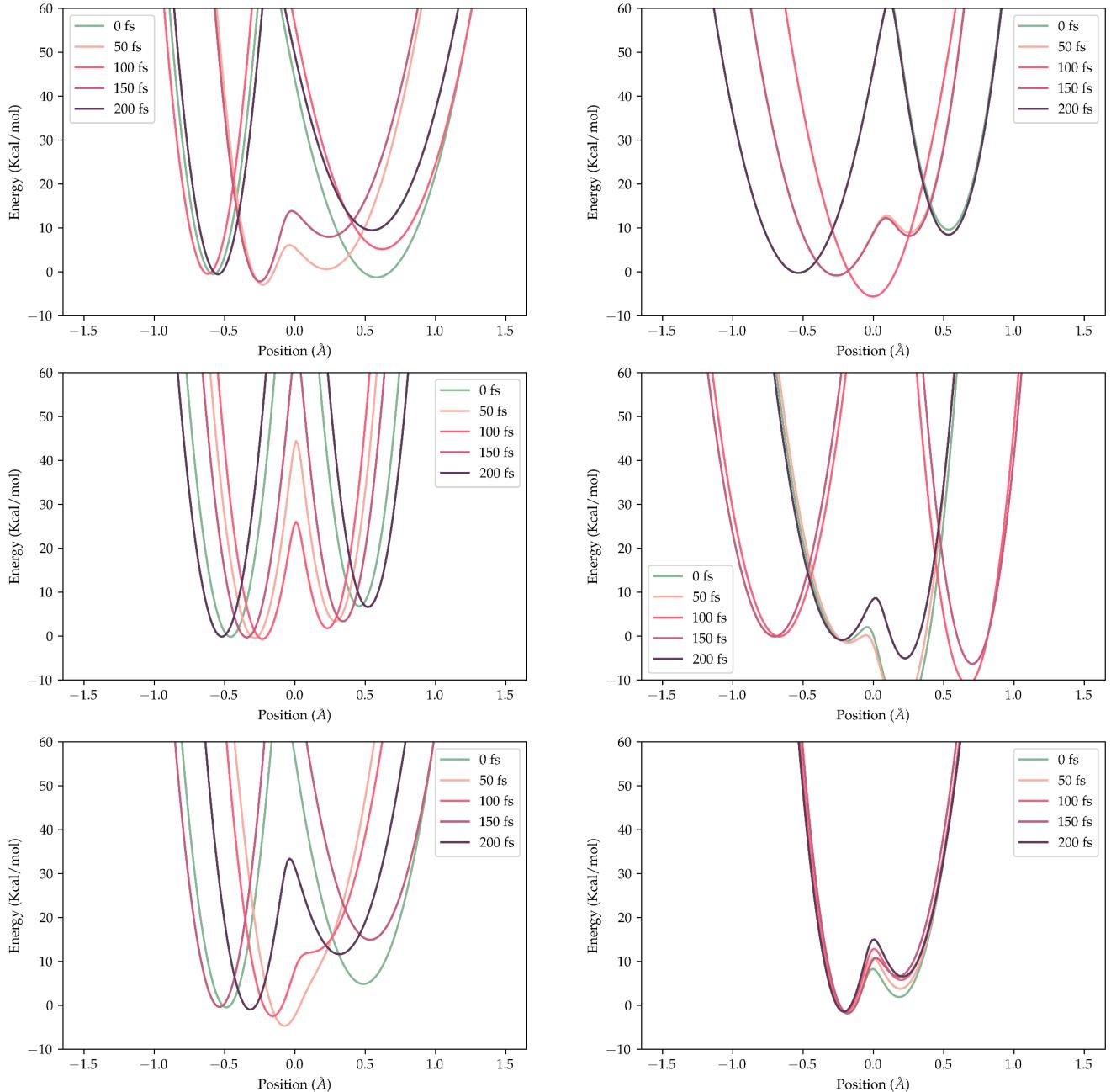
VARIABLE	VALOR
$V$	0.015936 [a.u]
$\omega_1$	0.0110287 [a.u]
$\omega_2$	0.0141751 [a.u]
$\lambda$	0.00755816 [a.u]
$X_{eq}$	0.00891848 [a.u]
$\omega_x$	0.000357994 [ $Jiffy^{-1}$ ]
$\theta_X$	5.7759 [rad]
$R_{eq}$	1.51438 [ $a_0$ ]
$R_0$	1.17396 [ $a_0$ ]
$\omega_R$	0.00103848 [au]
$\theta_R$	0.429406 [rad]
$m$	1836 [ $m_e$ ]

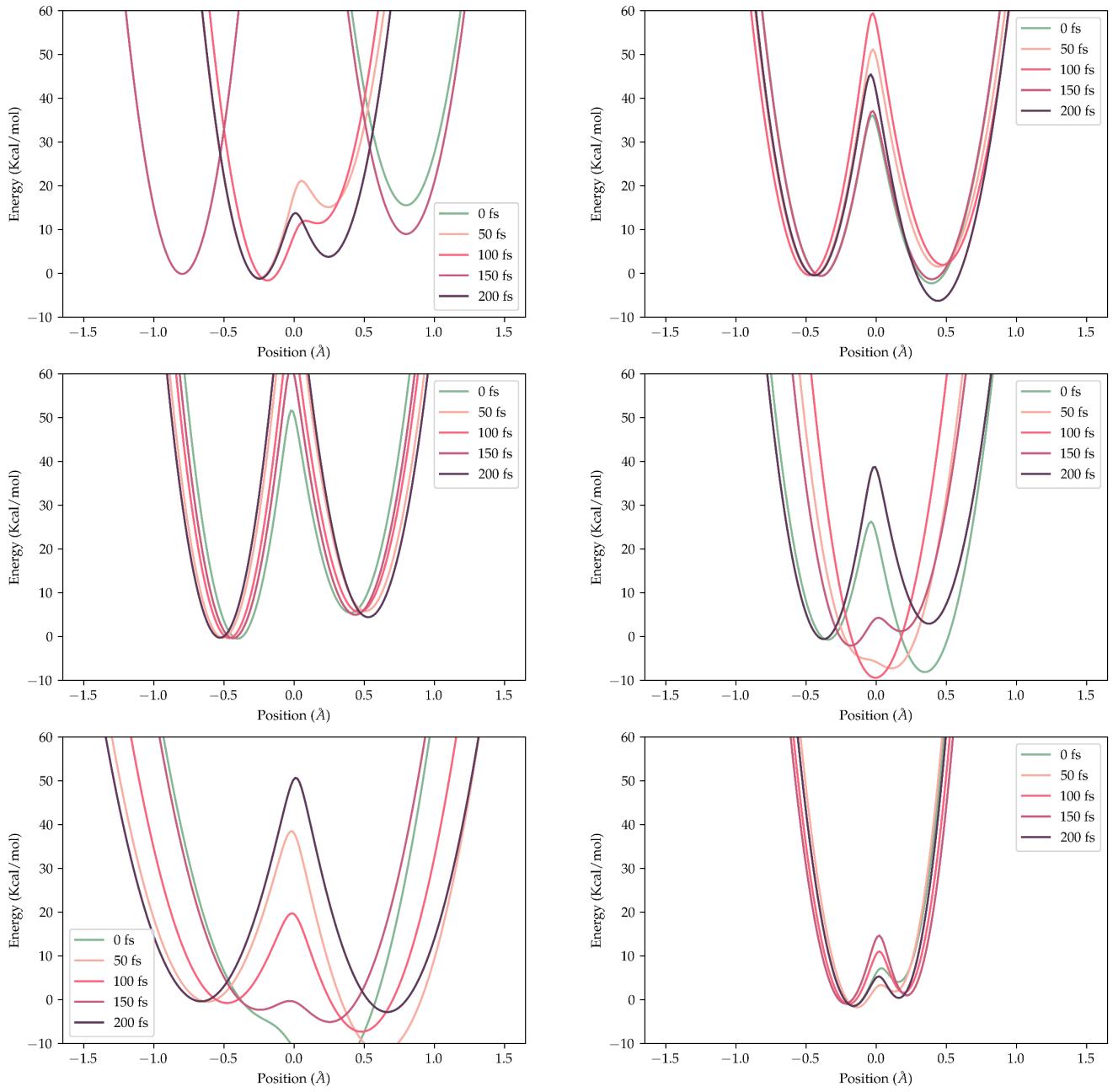
Tabla A.1: Valores de parámetros del potencial utilizados para generar la gráfica de la Figura 3.3

VARIABLE	VALOR
$V$	0.015936 [a.u.]
$\omega_1$	0.006834 – 0.018224 [a.u.]
$\omega_2$	0.006834 – 0.018224 [a.u.]
$\lambda$	0 – 0.015936 [a.u.]
$X_{eq}$	-0.015936 – 0.015936 [a.u.]
$\omega_x$	0.000357994 [ $Jiffy^{-1}$ ]
$\theta_X$	0 – $2\pi$ [rad]
$R_{eq}$	0.377945 – 1.889725 [ $a_0$ ]
$R_0$	0 – $2R_{eq}$ [ $a_0$ ]
$\omega_R$	0.0004556 – 0.0013668 [a.u.]
$\theta_R$	0 – $2\pi$ [rad]
$m$	1836 [ $m_e$ ]

Tabla A.2: Rango de valores de parámetros del potencial utilizados para generar trayectorias.

## A.3 EJEMPLOS DE POTENCIALES UTILIZADOS

Figura A.1: Ejemplos de potenciales utilizados  $V(r, t)$

Figura A.2: Ejemplos de potenciales utilizados  $V(r, t)$

## A.4 UNIDADES ATÓMICAS (A.U.)

UNIDADES ATÓMICAS	VALOR SI	NOMBRE (SÍMBOLO)
Masa $m_e$	9.10939(-31)	Masa del electrón
Carga $e$	1.602188(-19)	Carga del electrón
Momento angular $\hbar$	1.05457(-34)	Constante de Planck/ $2\pi$
Energía ( $m_e e^4 / \hbar^2$ )	4.35975(-18)	Hartree ( $H$ )
Longitud ( $\hbar^2 / m_e e^2$ )	5.29177(-11)	Radio de Bohr ( $a_0$ )
Tiempo ( $\hbar^3 / m_e e^4$ )	2.41888(-17)	Jiffy
Momento Dipolar Eléctrico ( $\hbar^2 / m_e e$ )	8.47836(-30)	2.541765 Debye ( $D$ ) units
Momento Dipolar Magnético ( $e\hbar / 2m_e$ )	9.27402(-24)	Magneton de Bohr ( $\mu_B$ )

Tabla A.3: Conversión de unidades atómicas a unidades SI.

Unidad	<i>a.u.</i>	<i>kcal/mol</i>	<i>eV</i>	<i>cm<sup>-1</sup></i>	<i>Hz</i>
<i>a.u.</i>	1	6.27510(2)	2.72114(1)	2.19475(5)	6.57968(15)
<i>kcal/mol</i>	1.59360(-3)	1	4.33641(-2)	3.49755(2)	1.04854(13)
<i>eV</i>	3.67493(-2)	2.30605(1)	1	8.06554(3)	2.41799(14)
<i>cm<sup>-1</sup></i>	4.55634(-6)	2.85914(-3)	1.23984(-4)	1	2.99792(10)
<i>Hz</i>	1.51983(-16)	9.5371(-14)	4.13567(-15)	3.33564(-11)	1

Tabla A.4: Conversión de energía (diversas unidades).

---

---

## BIBLIOGRAFÍA

- [1] Charu C. Aggarwal. *Neural Networks and Deep Learning*. Springer International Publishing, 2018. doi: [10.1007/978-3-319-94463-0](https://doi.org/10.1007/978-3-319-94463-0). URL: <https://doi.org/10.1007/978-3-319-94463-0>.
- [2] Tuguldur Kh. Begzjav **and** Hichem Eleuch. “Magnus expansion applied to a dissipative driven two-level system”. in: *Results in Physics* 17 (june 2020), page 103098. doi: [10.1016/j.rinp.2020.103098](https://doi.org/10.1016/j.rinp.2020.103098). URL: <https://doi.org/10.1016/j.rinp.2020.103098>.
- [3] François Chollet. *Deep Learning with Python*. Manning, november 2017. ISBN: 9781617294433.
- [4] Daniel T. Colbert **and** William H. Miller. “A novel discrete variable representation for quantum mechanical reactive scattering via the S-matrix Kohn method”. in: *The Journal of Chemical Physics* 96.3 (february 1992), pages 1982–1991. doi: [10.1063/1.462100](https://doi.org/10.1063/1.462100). URL: <https://doi.org/10.1063/1.462100>.
- [5] James T. Hynes Daniel Borgis. “Dynamical theory of proton tunneling transfer rates in solution: general formulation”. in: *Chemical Physics* 170 (1993), pages 315–346. doi: [10.1016/0301-0104\(93\)85117-Q](https://doi.org/10.1016/0301-0104(93)85117-Q). URL: [https://doi.org/10.1016/0301-0104\(93\)85117-Q](https://doi.org/10.1016/0301-0104(93)85117-Q).
- [6] Hassan Ismail Fawaz, Germain Forestier, Jonathan Weber, Lhassane Idoumghar **and** Pierre-Alain Muller. “Deep learning for time series classification: a review”. in: *Data Mining and Knowledge Discovery* 33.4 (march 2019), pages 917–963. doi: [10.1007/s10618-019-00619-1](https://doi.org/10.1007/s10618-019-00619-1). URL: <https://doi.org/10.1007/s10618-019-00619-1>.
- [7] Richard P. Feynman. *The Feynman Lectures On Physics Volume III: Quantum Mechanics* 3. 140522731 edition. volume III. The Feynman Lectures On Physics 03. Addison-Wesley Pub, 1977. ISBN: 0201020106.
- [8] Cutter Mary Ann Gardell. *The brain: Understanding neurobiology through the study of addiction*. Center for Curriculum Development, 2010.

- [9] Walther Gerlach **and** Otto Stern. "Der experimentelle Nachweis der Richtungsquantelung im Magnetfeld". in: *Zeitschrift f r Physik* 9.1 (**december 1922**), **pages** 349–352. ISSN: 1434-601X. DOI: [10.1007/BF01326983](https://doi.org/10.1007/BF01326983). URL: <http://dx.doi.org/10.1007/BF01326983>.
- [10] I. Goodfellow, Y. Bengio **and** A. Courville. *Deep Learning*. Adaptive computation and machine learning. MIT Press, 2016. ISBN: 9780262035613. URL: <https://books.google.co.in/books?id=Np9SDQAAQBAJ>.
- [11] David Gottlieb **and** Steven A. Orszag. *Numerical Analysis of Spectral Methods*. Society for Industrial **and** Applied Mathematics, **january 1977**. DOI: [10.1137/1.9781611970425](https://doi.org/10.1137/1.9781611970425). URL: <https://doi.org/10.1137/1.9781611970425>.
- [12] David J Griffiths. *Introduction to Quantum Mechanics*. Upper Saddle River, NJ: Pearson, **august 1994**.
- [13] Sepp Hochreiter **and** J rgen Schmidhuber. "Long Short-Term Memory". in: *Neural Computation* 9.8 (**november 1997**), **pages** 1735–1780. DOI: [10.1162/neco.1997.9.8.1735](https://doi.org/10.1162/neco.1997.9.8.1735). URL: <https://doi.org/10.1162/neco.1997.9.8.1735>.
- [14] Kurt Hornik, Maxwell Stinchcombe **and** Halbert White. "Multilayer feedforward networks are universal approximators". in: *Neural Networks* 2.5 (**january 1989**), **pages** 359–366. DOI: [10.1016/0893-6080\(89\)90020-8](https://doi.org/10.1016/0893-6080(89)90020-8). URL: [https://doi.org/10.1016/0893-6080\(89\)90020-8](https://doi.org/10.1016/0893-6080(89)90020-8).
- [15] V Jelic **and** F Marsiglio. "The double-well potential in quantum mechanics: a simple, numerically exact formulation". in: *European Journal of Physics* 33.6 (**september 2012**), **pages** 1651–1666. DOI: [10.1088/0143-0807/33/6/1651](https://doi.org/10.1088/0143-0807/33/6/1651). URL: <https://doi.org/10.1088/0143-0807/33/6/1651>.
- [16] Andrej Karpathy, Justin Johnson **and** Li Fei-Fei. *Visualizing and Understanding Recurrent Networks*. 2015. DOI: [10.48550/ARXIV.1506.02078](https://arxiv.org/abs/1506.02078). URL: <https://arxiv.org/abs/1506.02078>.
- [17] Joshua P. Layfield **and** Sharon Hammes-Schiffer. "Hydrogen Tunneling in Enzymes and Biomimetic Models". in: *Chemical Reviews* 114.7 (2014). PMID: 24359189, **pages** 3466–3494. DOI: [10.1021/cr400400p](https://doi.org/10.1021/cr400400p). eprint: <https://doi.org/10.1021/cr400400p>. URL: <https://doi.org/10.1021/cr400400p>.
- [18] J. C. Light, I. P. Hamilton **and** J. V. Lill. "Generalized discrete variable approximation in quantum mechanics". in: *The Journal of Chemical Physics* 82.3 (**february 1985**), **pages** 1400–1409. DOI: [10.1063/1.448462](https://doi.org/10.1063/1.448462). URL: <https://doi.org/10.1063/1.448462>.

- [19] John C. Light **and** Tucker Carrington. *Discrete-Variable Representations and their Utilization*. **january** 2000. DOI: [10.1002/9780470141731.ch4](https://doi.org/10.1002/9780470141731.ch4). URL: <https://doi.org/10.1002/9780470141731.ch4>.
- [20] Robert G. Littlejohn, Matthew Cargo, Tucker Carrington, Kevin A. Mitchell **and** Bill Poirier. “A general framework for discrete variable representation basis sets”. **in:** *The Journal of Chemical Physics* 116.20 (**may** 2002), **pages** 8691–8703. DOI: [10.1063/1.1473811](https://doi.org/10.1063/1.1473811). URL: <https://doi.org/10.1063/1.1473811>.
- [21] Ilya Loshchilov **and** Frank Hutter. *Decoupled Weight Decay Regularization*. 2017. DOI: [10.48550/ARXIV.1711.05101](https://arxiv.org/abs/1711.05101). URL: <https://arxiv.org/abs/1711.05101>.
- [22] Kevin P Murphy. *Machine Learning: A Probabilistic Perspective*. Adaptive Computation and Machine Learning series. London, England: MIT Press, **august** 2012.
- [23] Daniel Neil, Michael Pfeiffer **and** Shih-Chii Liu. *Phased LSTM: Accelerating Recurrent Network Training for Long or Event-based Sequences*. 2016. DOI: [10.48550/ARXIV.1610.09513](https://arxiv.org/abs/1610.09513). URL: <https://arxiv.org/abs/1610.09513>.
- [24] Christopher Olah. *Understanding LSTM Networks*. <http://colah.github.io/posts/2015-08-Understanding-LSTMs/>. Consultada el 02 de septiembre de 2023. 2015.
- [25] Luigi E. Picasso, Luciano Bracci **and** Emilio d'Emilio. “Perturbation Theory in Quantum Mechanics”. **in:** *Mathematics of Complexity and Dynamical Systems*. Springer New York, 2012, **pages** 1351–1375. DOI: [10.1007/978-1-4614-1806-1\\_85](https://doi.org/10.1007/978-1-4614-1806-1_85). URL: [https://doi.org/10.1007/978-1-4614-1806-1\\_85](https://doi.org/10.1007/978-1-4614-1806-1_85).
- [26] Haşim Sak, Andrew Senior **and** Françoise Beaufays. *Long Short-Term Memory Based Recurrent Neural Network Architectures for Large Vocabulary Speech Recognition*. 2014. DOI: [10.48550/ARXIV.1402.1128](https://arxiv.org/abs/1402.1128). URL: <https://arxiv.org/abs/1402.1128>.
- [27] J. J. Sakurai. *Modern Quantum Mechanics*. Rev. ed. Addison-Wesley, 1994. ISBN: 9780201539295.
- [28] K. T. Schütt, H. E. Sauceda, P.-J. Kindermans, A. Tkatchenko **and** K.-R. Müller. “SchNet – A deep learning architecture for molecules and materials”. **in:** *The Journal of Chemical Physics* 148.24 (**march** 2018). DOI: [10.1063/1.5019779](https://doi.org/10.1063/1.5019779). URL: <https://doi.org/10.1063/1.5019779>.

- [29] Maxim Secor, Alexander V. Soudackov **and** Sharon Hammes-Schiffer. "Artificial Neural Networks as Propagators in Quantum Dynamics". **in:** *The Journal of Physical Chemistry Letters* 12.43 (2021). PMID: 34704767, **pages** 10654–10662. doi: [10.1021/acs.jpclett.1c03117](https://doi.org/10.1021/acs.jpclett.1c03117). eprint: <https://doi.org/10.1021/acs.jpclett.1c03117>. URL: <https://doi.org/10.1021/acs.jpclett.1c03117>.
- [30] David J. Tannor. *Introduction to Quantum Mechanics: A Time-Dependent Perspective*. University Science Books, 2006. ISBN: 1891389238.
- [31] Ivan Vasilev, Daniel Slater, Gianmario Spacagna, Peter Roelants **and** Valentino Zocca. *Python deep learning: Exploring deep learning techniques and neural network architectures with pytorch, Keras, and tensorflow*. Packt Publishing, 2019.
- [32] Bipeng Wang, Ludwig Winkler, Yifan Wu, Klaus-Robert Müller, Huziel E. Sauceda **and** Oleg V. Prezhdo. "Interpolating Nonadiabatic Molecular Dynamics Hamiltonian with Bidirectional Long Short-Term Memory Networks". **in:** *The Journal of Physical Chemistry Letters* 14.31 (august 2023), **pages** 7092–7099. doi: [10.1021/acs.jpclett.3c01723](https://doi.org/10.1021/acs.jpclett.3c01723). URL: <https://doi.org/10.1021/acs.jpclett.3c01723>.