dirac_solver

Generado por Doxygen 1.9.4

1 README 1

2 Plan del Proyecto para un Solucionador de la Ecuación de Dirac 3

2.1 1. Partícula Libre . . . 3

2.1.1 Descripción del Problema . . . 3

2.1.2 Parámetros de Entrada . . . 3

2.1.3 Salidas . . . 3

2.2 2. Pozo de Potencial Infinito . . . 3

2.2.1 Descripción del Problema . . . 4

2.2.2 Parámetros de Entrada . . . 4

2.2.3 Salidas . . . 4

2.3 3. Potencial a Trozos (Efecto Klein) . . . 4

2.3.1 Descripción del Problema . . . 4

2.3.2 Parámetros de Entrada . . . 4

2.3.3 Salidas . . . 4

2.4 4. Potencial Central (Interacción de Coulomb) . . . 4

2.4.1 Descripción del Problema . . . 4

2.4.2 Parámetros de Entrada . . . 5

2.4.3 Salidas . . . 5

2.5 5. Potencial de Yukawa . . . 5

2.5.1 Descripción del Problema . . . 5

2.5.2 Parámetros de Entrada . . . 5

2.5.3 Salidas . . . 5

3 Estructura del Proyecto dirac_solver 7

3.1 Esquema Visual . . . 7

3.2 Raíz del Proyecto . . . 8

3.3 <tt>src/dirac_solver/</tt> . . . 8

3.3.1 1. <tt>solvers/</tt> Núcleo numérico . . . 8

3.3.2 2. <tt>physics/</tt> Definiciones físicas . . . 8

3.3.3 3. <tt>potentials/</tt> Potenciales . . . 8

3.3.4 4. <tt>geometry/</tt> Geometrías y dominios . . . 9

3.3.5 5. <tt>conditions/</tt> Condiciones . . . 9

3.3.6 6. <tt>io/</tt> Entrada/Salida . . . 9

3.3.7 7. <tt>visualization/</tt> Gráficos y análisis . . . 9

3.3.8 8. <tt>utils/</tt> Utilidades generales . . . 9

3.4 <tt>tests/</tt> . . . 10

3.5 <tt>examples/</tt> . . . 10

3.6 <tt>docs/</tt> . . . 10

4 Dirac-Solver: Solucionador numérico de la Ecuación de Dirac. 11

4.1 Documentación . . . 11

4.2 Instalación de la libreria . . . 11

5 Indice de namespaces 13
5.1 Lista de 'namespaces' . . . 13

6 Indice jerárquico 15
6.1 Jerarquía de la clase . . . 15

7 Índice de clases 17
7.1 Lista de clases . . . 17

8 Indice de archivos 19
8.1 Lista de archivos . . . 19

9 Documentación de namespaces 21
9.1 Referencia del Namespace check_package . . . 21
9.1.1 Documentación de las variables . . . 21
9.1.1.1 base_path . . . 21
9.1.1.2 encoding . . . 21
9.1.1.3 init_path . . . 21
9.2 Referencia del Namespace dirac_solver . . . 21
9.3 Referencia del Namespace dirac_solver.core . . . 21
9.4 Referencia del Namespace dirac_solver.geometry . . . 22
9.5 Referencia del Namespace dirac_solver.initial_state . . . 22

10 Documentación de las clases 23
10.1 Referencia de la Clase dirac_solver.core.DiracSolver . . . 23
10.1.1 Descripción detallada . . . 23
10.1.2 Documentación de las funciones miembro . . . 23
10.1.2.1 __initial__() . . . 23
10.1.2.2 plot_probability_density() . . . 23
10.1.2.3 run_simulation() . . . 23
10.2 Referencia de la Clase dirac_solver.geometry.Geometry . . . 24
10.2.1 Descripción detallada . . . 24
10.3 Referencia de la Clase dirac_solver.initial_state.InitialState . . . 24
10.3.1 Descripción detallada . . . 24
10.4 Referencia de la Clase dirac_solver.geometry.Interval1D . . . 25
10.4.1 Descripción detallada . . . 25
10.4.2 Documentación del constructor y destructor . . . 25
10.4.2.1 __init__() . . . 25
10.4.3 Documentación de los datos miembro . . . 26
10.4.3.1 max . . . 26
10.4.3.2 min . . . 26
10.5 Referencia de la Clase dirac_solver.initial_state.PlaneWave . . . 26
10.5.1 Descripción detallada . . . 27
10.5.2 Documentación del constructor y destructor . . . 27

10.5.2.1 __init__() 27
10.5.3 Documentación de las funciones miembro 27
10.5.3.1 create_state() 27

11 Documentación de archivos **29**
11.1 Referencia del Archivo dirac_solver/check_package.py 29
11.2 Referencia del Archivo dirac_solver/src/dirac_solver/__init__.py 29
11.3 Referencia del Archivo dirac_solver/src/dirac_solver/core.py 29
11.4 Referencia del Archivo dirac_solver/src/dirac_solver/geometry.py 29
11.5 Referencia del Archivo dirac_solver/src/dirac_solver/initial_state.py 30
11.6 Referencia del Archivo planning/problems-to-solve.md 30
11.7 Referencia del Archivo planning/structure.md 30
11.8 Referencia del Archivo dirac_solver/README.md 30
11.9 Referencia del Archivo README.md 30

Índice alfabético **31**

Capítulo 1

README

Solo es para hacer la prueba de la instalacíon.

para instalar, con un entorno activo instalar con

pip install -e . dentro de esta carpeta

Capítulo 2

Plan del Proyecto para un Solucionador de la Ecuación de Dirac

Este documento describe el plan del proyecto para un solucionador de la ecuación de Dirac. El solucionador manejará la ecuación $(i\gamma^\mu \partial_\mu - m)\psi=0$ en unidades naturales, donde el bispinor $\psi \in \mathbb{C}^4$ es un vector columna de valores complejos. Las siguientes secciones detallan los problemas a abordar, junto con sus respectivos requisitos computacionales.

2.1. 1. Partícula Libre

El caso de la partícula libre es el problema fundamental, sirviendo como referencia para la funcionalidad central del solucionador.

2.1.1. Descripción del Problema

Resolver la ecuación de Dirac para una partícula en ausencia de un potencial externo. El desarrollo inicial se centrará en el caso unidimensional (1+1D), ya que cualquier movimiento de una partícula libre puede ser transformado a esta configuración mediante un boost de Lorentz.

2.1.2. Parámetros de Entrada

- **Masa de la Partícula** (m)
- **Momento Inicial** (p)
- **Distribución Espacial**: Los estados iniciales incluirán tanto una simple onda plana como un paquete de ondas Gaussiano más físicamente realista.
- **Estado de Espín**: La configuración inicial del espín del bispinor.

2.1.3. Salidas

- **Densidades de Probabilidad y Corriente**: La evolución temporal de la densidad de probabilidad $\rho = \psi^\dagger \psi$ y la densidad de corriente $J = \psi^\dagger \alpha \psi$.
- **Valores Esperados**: Los valores esperados para la energía, el momento y la posición a lo largo del tiempo, que se utilizarán para observar fenómenos relativistas como la Zitterbewegung.

2.2. 2. Pozo de Potencial Infinito

Este problema introduce condiciones de contorno, probando la capacidad del solucionador para manejar el confinamiento.

2.2.1. Descripción del Problema

Resolver la ecuación de Dirac para una partícula confinada dentro de un pozo de potencial infinito de ancho L. El bispinor ψ debe anularse en las paredes del pozo.

2.2.2. Parámetros de Entrada

- **Ancho del Pozo** (L)
- **Masa de la Partícula** (m) y Estado de Espín Inicial.

2.2.3. Salidas

- **Niveles de Energía Cuantizados**: Los valores propios de energía discretos del sistema.
- **Autofunciones**: Las funciones de onda del bispinor correspondientes a cada nivel de energía.
- **Evolución Temporal**: La evolución temporal de la densidad de probabilidad para un estado inicial, demostrando el comportamiento oscilatorio dentro del pozo.

2.3. 3. Potencial a Trozos (Efecto Klein)

Este caso es crucial para demostrar el fenómeno relativista del Efecto Klein.

2.3.1. Descripción del Problema

Resolver el comportamiento de una partícula que encuentra un escalón de potencial. El objetivo es demostrar que una partícula relativista puede atravesar una barrera de potencial incluso cuando su energía es menor que la altura de la barrera, un resultado clásicamente imposible.

2.3.2. Parámetros de Entrada

- **Altura del Potencial** (V_0)
- **Energía de la Partícula** (E) y Momento Inicial (p)
- **Masa de la Partícula** (m) y Estado de Espín Inicial.

2.3.3. Salidas

- **Coeficientes de Reflexión (R) y Transmisión (T)**: La salida principal, que demuestra que la transmisión puede ocurrir incluso cuando $V_0 > E+m$.
- **Visualización de la Función de Onda**: Un gráfico de la función de onda del bispinor que muestra los componentes entrante, reflejado y transmitido.

2.4. 4. Potencial Central (Interacción de Coulomb)

Este problema es una aplicación fundamental de la ecuación de Dirac, abordando la estructura fina del átomo de hidrógeno.

2.4.1. Descripción del Problema

Resolver la ecuación de Dirac para un electrón en un potencial central, específicamente el potencial de Coulomb $V(r) = -Z\alpha/r$.

2.4.2. Parámetros de Entrada

- **Carga Central** (Z) (p.ej., $Z=1$ para el hidrógeno).
- **Masa de la Partícula** (m) (la masa del electrón).
- **Número Cuántico del Momento Angular** (j): La solución se maneja mejor en ă coordenadas esféricas.

2.4.3. Salidas

- **Niveles de Energía Cuantizados**: Los valores propios de energía discretos, que deben coincidir con la fórmula de la estructura fina de Sommerfeld.
- **Funciones de Onda**: Los componentes radiales de la función de onda del bispinor.
- **Desdoblamiento Espín-Órbita**: El solucionador debe reproducir correctamente el desdoblamiento de los niveles de energía debido a la interacción espín-órbita.

2.5. 5. Potencial de Yukawa

Este problema proporciona una generalización crucial del potencial de Coulomb, relevante para la física nuclear.

2.5.1. Descripción del Problema

Resolver el comportamiento de una partícula en un potencial de Yukawa de corto alcance $V(r) = -g \frac{e^{-Mr}}{r}$.

2.5.2. Parámetros de Entrada

- **Constante de Acoplamiento** (g) y **Masa de la Partícula Mediadora** (M): Estos parámetros definen la fuerza y el alcance del potencial.
- **Masa de la Partícula** (m) y Estado Inicial.

2.5.3. Salidas

- **Energías de los Estados Ligados**: Los valores propios de energía discretos para los estados ligados.
- **Funciones de Onda**: Una visualización de las funciones de onda del bispinor, mostrando cómo difieren del caso de Coulomb debido al decaimiento exponencial del potencial.
- **Verificación de Consistencia**: La salida debe demostrar que los resultados convergen al caso de Coulomb cuando la masa de la partícula mediadora $M \to 0$.

Capítulo 3

Estructura del Proyecto dirac_solver

Este documento describe la estructura modular del proyecto `dirac_solver`, inspirada en el diseño de librerías como **DeepXDE**. La idea es mantener una organización clara, extensible y reutilizable para resolver la ecuación de Dirac en diferentes escenarios físicos.

3.1. Esquema Visual

```
dirac_solver/
 pyproject.toml
 README.md
 LICENSE
 src/
    dirac_solver/
        __init__.py

        solvers/
           __init__.py
           time_integrator.py
           eigensolver.py
           observables.py

        physics/
           __init__.py
           gamma_matrices.py
           spinors.py
           relativistic_identities.py

        potentials/
           __init__.py
           base.py
           coulomb.py
           yukawa.py
           step.py
           infinite_well.py
           custom.py

        geometry/
           __init__.py
           interval.py
           radial.py
           grid.py

        conditions/
           __init__.py
           initial.py
           boundary.py

        io/
           __init__.py
           saver.py
           loader.py
           config.py

        visualization/
           __init__.py
           plot_wavefunction.py
           plot_density.py
           animations.py

        utils/
            __init__.py
            math_ops.py
            validators.py
```

```
            constants.py

tests/
   test_potentials.py
   test_geometry.py
   test_boundary.py
   test_solver_free.py
   test_solver_infinite_well.py
   test_solver_klein.py
   test_solver_coulomb.py
   test_solver_yukawa.py

examples/
   free_particle.py
   infinite_well.py
   klein_paradox.py
   hydrogen_atom.py
   yukawa_bound_states.py

docs/
    index.md
    api_reference.md
    theory_background.md
```

3.2. Raíz del Proyecto

- **pyproject.toml**
  Archivo de configuración para empaquetar e instalar la librería (`pip install dirac_solver`). Define dependencias, metadatos y backend de construcción.
- **README.md**
  Introducción general al proyecto, instalación, ejemplos rápidos de uso y enlaces a documentación.
- **LICENSE**
  Licencia del proyecto (ejemplo: MIT, GPL, BSD).

3.3. <tt>src/dirac_solver/</tt>

Contiene el **código fuente principal** de la librería.

3.3.1. 1. <tt>solvers/</tt> Núcleo numérico

- **time_integrator.py**: Métodos de evolución temporal (CrankNicolson, Split-Operator, RungeKutta).
- **eigensolver.py**: Resolución de problemas estacionarios (autovalores/autofunciones, bound states).
- **observables.py**: Cálculo de valores esperados: energía, posición, momento, corrientes.

3.3.2. 2. <tt>physics/</tt> Definiciones físicas

- **gamma_matrices.py**: Representaciones de matrices de Dirac en distintas dimensiones.
- **spinors.py**: Construcción de espinores iniciales (partícula libre, polarización de espín).
- **relativistic_identities.py**: Utilidades analíticas (Zitterbewegung, relaciones de conmutación).

3.3.3. 3. <tt>potentials/</tt> Potenciales

- **base.py**: Clase abstracta `Potential` para herencia.
- **coulomb.py**: Potencial de Coulomb (átomo de hidrógeno).

- **yukawa.py**: Potencial de Yukawa (interacciones de corto alcance).
- **step.py**: Escalón de potencial (Efecto Klein).
- **infinite_well.py**: Pozo de potencial infinito.
- **custom.py**: Permite definir potenciales personalizados por el usuario.

3.3.4. 4. <tt>geometry/</tt> Geometrías y dominios

- **interval.py**: Intervalo unidimensional con discretización uniforme.
- **radial.py**: Coordenada radial (casos centrales como Coulomb/Yukawa).
- **grid.py**: Utilidades para crear mallas, espaciamientos y operadores de diferencias finitas.

3.3.5. 5. <tt>conditions/</tt> Condiciones

- **initial.py**: Estados iniciales (onda plana, paquete gaussiano, superposición).
- **boundary.py**: Condiciones de frontera (Dirichlet, Neumann, absorbentes).

3.3.6. 6. <tt>io/</tt> Entrada/Salida

- **saver.py**: Guardar snapshots de la evolución temporal.
- **loader.py**: Cargar simulaciones previas.
- **config.py**: Configuración desde archivos YAML/JSON (ejecución reproducible).

3.3.7. 7. <tt>visualization/</tt> Gráficos y análisis

- **plot_wavefunction.py**: Visualización de las componentes del bispinor.
- **plot_density.py**: Densidades de probabilidad y corrientes.
- **animations.py**: Animaciones temporales (propagación, confinamiento).

3.3.8. 8. <tt>utils/</tt> Utilidades generales

- **math_ops.py**: Operaciones auxiliares numéricas y algebraicas.
- **validators.py**: Validación de entradas y parámetros físicos.
- **constants.py**: Constantes físicas (, c, , etc.).
- **coolsftuf.py**: Comentarios chistosos.

3.4. <tt>tests/</tt>

Tests unitarios organizados por módulos. Verifican consistencia, conservación de probabilidad y reproducción de fenómenos físicos.

- **test_potentials.py**: Comprueba la implementación de los potenciales.
- **test_geometry.py**: Valida discretizaciones espaciales.
- **test_boundary.py**: Verifica condiciones de frontera.
- **test_solver_free.py**: Propagación de partícula libre.
- **test_solver_infinite_well.py**: Cuantización en pozo infinito.
- **test_solver_klein.py**: Coeficientes de reflexión y transmisión (efecto Klein).
- **test_solver_coulomb.py**: Niveles del átomo de hidrógeno relativista.
- **test_solver_yukawa.py**: Estados ligados en potencial de Yukawa.

3.5. <tt>examples/</tt>

Ejemplos reproducibles de uso del solver en diferentes escenarios.

- **free_particle.py**: Partícula libre en 1+1D.
- **infinite_well.py**: Cuantización en un pozo infinito.
- **klein_paradox.py**: Simulación del efecto Klein.
- **hydrogen_atom.py**: Estructura fina del átomo de hidrógeno.
- **yukawa_bound_states.py**: Estados ligados en potencial de Yukawa.

3.6. <tt>docs/</tt>

Documentación del proyecto.

- **index.md**: Página principal de la documentación.
- **api_reference.md**: Referencia de clases, funciones y módulos.
- **theory_background.md**: Fundamentos teóricos de la ecuación de Dirac y fenómenos relativistas.

Capítulo 4

Dirac-Solver: Solucionador numérico de la Ecuación de Dirac.

[aqui va una interesante introducción]

4.1. Documentación

Para compilar la documentación, en el directorio principal ejecutar:
-> Doxigen
Esto creará los subdirectorios ./docs/html y ./docs/latex
Para obtener el .pdf de latex, desde el directorio principal:
-> cd ./docs/html -> make

4.2. Instalación de la libreria

En el entorno principal ejecutar:
-> cd ./dirac_solver/ -> pip install -e .

Capítulo 5

Indice de namespaces

5.1. Lista de 'namespaces'

Lista de los 'namespaces', con una breve descripción:

check_package 21
dirac_solver 21
dirac_solver.core 21
dirac_solver.geometry 22
dirac_solver.initial_state 22

Capítulo 6

Indice jerárquico

6.1. Jerarquía de la clase

Esta lista de herencias esta ordenada aproximadamente por orden alfabético:

dirac_solver.core.DiracSolver 23
dirac_solver.geometry.Geometry 24
  dirac_solver.geometry.Interval1D 25
dirac_solver.initial_state.InitialState 24
  dirac_solver.initial_state.PlaneWave 26

Capítulo 7

Índice de clases

7.1. Lista de clases

Lista de las clases, estructuras, uniones e interfaces con una breve descripción:

dirac_solver.core.DiracSolver 23
dirac_solver.geometry.Geometry 24
dirac_solver.initial_state.InitialState 24
dirac_solver.geometry.Interval1D 25
dirac_solver.initial_state.PlaneWave 26

Capítulo 8

Indice de archivos

8.1. Lista de archivos

Lista de todos los archivos con descripciones breves:

dirac_solver/check_package.py 29
dirac_solver/src/dirac_solver/__init__.py 29
dirac_solver/src/dirac_solver/core.py 29
dirac_solver/src/dirac_solver/geometry.py 29
dirac_solver/src/dirac_solver/initial_state.py 30

Capítulo 9

Documentación de namespaces

9.1. Referencia del Namespace check_package

Variables

- string base_path = "src/dirac_solver"
- init_path = os.path.join(base_path, "__init__.py")
- encoding

9.1.1. Documentación de las variables

9.1.1.1. base_path

```
string check_package.base_path = "src/dirac_solver"
```

9.1.1.2. encoding

```
check_package.encoding
```

9.1.1.3. init_path

```
check_package.init_path = os.path.join(base_path, "__init__.py")
```

9.2. Referencia del Namespace dirac_solver

Namespaces

- namespace core
- namespace geometry
- namespace initial_state

9.3. Referencia del Namespace dirac_solver.core

Clases

- class DiracSolver

9.4. Referencia del Namespace dirac_solver.geometry

Clases

- class Geometry
- class Interval1D

9.5. Referencia del Namespace dirac_solver.initial_state

Clases

- class InitialState
- class PlaneWave

Capítulo 10

Documentación de las clases

10.1. Referencia de la Clase dirac_solver.core.DiracSolver

Métodos públicos

- def __initial__ (self)
- def run_simulation (self)
- def plot_probability_density (self)

10.1.1. Descripción detallada

```
this is the principal class
```

10.1.2. Documentación de las funciones miembro

10.1.2.1. __initial__()

```
def dirac_solver.core.DiracSolver.__initial__ (
              self )
```

```
make the system (initial state + other parameters + geometry)
```

10.1.2.2. plot_probability_density()

```
def dirac_solver.core.DiracSolver.plot_probability_density (
              self )
```

```
here calculate and plot probability density. also is posible only calculate and save
```

10.1.2.3. run_simulation()

```
def dirac_solver.core.DiracSolver.run_simulation (
              self )
```

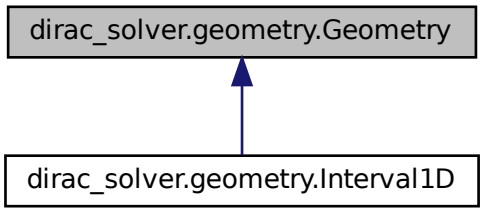
```
here the numerical method is implemented
```

La documentación para esta clase fue generada a partir del siguiente fichero:

- dirac_solver/src/dirac_solver/core.py

10.2. Referencia de la Clase dirac_solver.geometry.Geometry

Diagrama de herencias de dirac_solver.geometry.Geometry



10.2.1. Descripción detallada

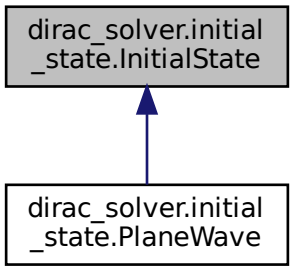
```
basis class for all geometrys
```

La documentación para esta clase fue generada a partir del siguiente fichero:

- dirac_solver/src/dirac_solver/geometry.py

10.3. Referencia de la Clase dirac_solver.initial_state.InitialState

Diagrama de herencias de dirac_solver.initial_state.InitialState



10.3.1. Descripción detallada

```
basis class for all states
```

La documentación para esta clase fue generada a partir del siguiente fichero:

- dirac_solver/src/dirac_solver/initial_state.py

10.4. Referencia de la Clase dirac_solver.geometry.Interval1D

Diagrama de herencias de dirac_solver.geometry.Interval1D

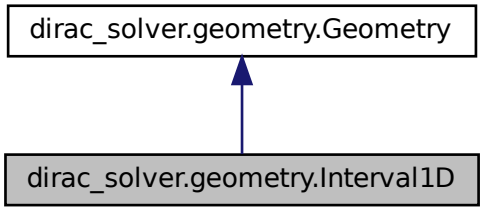
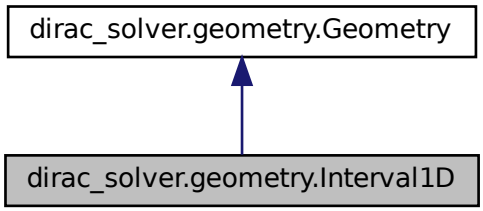


Diagrama de colaboración para dirac_solver.geometry.Interval1D:



Métodos públicos

- def __init__ (self, x_min, x_max)

Atributos públicos

- min
- max

10.4.1. Descripción detallada

```
1D geometry define in a interval L
```

10.4.2. Documentación del constructor y destructor

10.4.2.1. __init__()

```
def dirac_solver.geometry.Interval1D.__init__ (
            self,
            x_min,
            x_max )
```

10.4.3. Documentación de los datos miembro

10.4.3.1. max

`dirac_solver.geometry.Interval1D.max`

10.4.3.2. min

`dirac_solver.geometry.Interval1D.min`

La documentación para esta clase fue generada a partir del siguiente fichero:

- dirac_solver/src/dirac_solver/geometry.py

10.5. Referencia de la Clase dirac_solver.initial_state.PlaneWave

Diagrama de herencias de dirac_solver.initial_state.PlaneWave

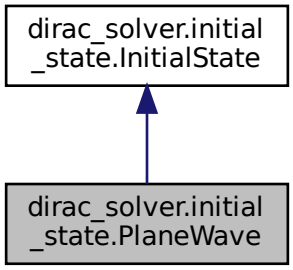
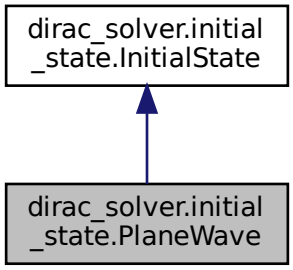


Diagrama de colaboración para dirac_solver.initial_state.PlaneWave:



Métodos públicos

- def __init__ (self)
- def create_state (self)

10.5.1. Descripción detallada

```
make planewave initialstate (for 1+1D free particle dirac)
```

10.5.2. Documentación del constructor y destructor

10.5.2.1. __init__()

```
def dirac_solver.initial_state.PlaneWave.__init__ (
              self )
```

10.5.3. Documentación de las funciones miembro

10.5.3.1. create_state()

```
def dirac_solver.initial_state.PlaneWave.create_state (
              self )
```

```
for create state
```

La documentación para esta clase fue generada a partir del siguiente fichero:

- dirac_solver/src/dirac_solver/initial_state.py

Capítulo 11

Documentación de archivos

11.1. Referencia del Archivo dirac_solver/check_package.py

Namespaces

- namespace check_package

Variables

- string check_package.base_path = "src/dirac_solver"
- check_package.init_path = os.path.join(base_path, "__init__.py")
- check_package.encoding

11.2. Referencia del Archivo dirac_solver/src/dirac_solver/__init__.py

Namespaces

- namespace dirac_solver

11.3. Referencia del Archivo dirac_solver/src/dirac_solver/core.py

Clases

- class dirac_solver.core.DiracSolver

Namespaces

- namespace dirac_solver
- namespace dirac_solver.core

11.4. Referencia del Archivo dirac_solver/src/dirac_solver/geometry.py

Clases

- class dirac_solver.geometry.Geometry
- class dirac_solver.geometry.Interval1D

Namespaces

- namespace dirac_solver
- namespace dirac_solver.geometry

11.5. Referencia del Archivo dirac_solver/src/dirac_solver/initial_state.py

Clases

- class dirac_solver.initial_state.InitialState
- class dirac_solver.initial_state.PlaneWave

Namespaces

- namespace dirac_solver
- namespace dirac_solver.initial_state

11.6. Referencia del Archivo planning/problems-to-solve.md

11.7. Referencia del Archivo planning/structure.md

11.8. Referencia del Archivo dirac_solver/README.md

11.9. Referencia del Archivo README.md

Índice alfabético

__init__
    dirac_solver.geometry.Interval1D, 25
    dirac_solver.initial_state.PlaneWave, 27
__initial__
    dirac_solver.core.DiracSolver, 23

base_path
    check_package, 21

check_package, 21
    base_path, 21
    encoding, 21
    init_path, 21
create_state
    dirac_solver.initial_state.PlaneWave, 27

dirac_solver, 21
dirac_solver.core, 21
dirac_solver.core.DiracSolver, 23
    __initial__, 23
    plot_probability_density, 23
    run_simulation, 23
dirac_solver.geometry, 22
dirac_solver.geometry.Geometry, 24
dirac_solver.geometry.Interval1D, 25
    __init__, 25
    max, 26
    min, 26
dirac_solver.initial_state, 22
dirac_solver.initial_state.InitialState, 24
dirac_solver.initial_state.PlaneWave, 26
    __init__, 27
    create_state, 27
dirac_solver/check_package.py, 29
dirac_solver/README.md, 30
dirac_solver/src/dirac_solver/__init__.py, 29
dirac_solver/src/dirac_solver/core.py, 29
dirac_solver/src/dirac_solver/geometry.py, 29
dirac_solver/src/dirac_solver/initial_state.py, 30

encoding
    check_package, 21

init_path
    check_package, 21

max
    dirac_solver.geometry.Interval1D, 26
min
    dirac_solver.geometry.Interval1D, 26

planning/problems-to-solve.md, 30
planning/structure.md, 30
plot_probability_density
    dirac_solver.core.DiracSolver, 23

README.md, 30
run_simulation
    dirac_solver.core.DiracSolver, 23