Desbloqueando Nuevas Herramientas para Estudiantes de Ingeniería: Análisis de una Antena Dipolo mediante PyAEDT

Celia Gómez Molina [(1)], Samuel López [(1)], Susannah Cooke [(2)].
celia.gomezmolina@ansys.com, samuel.lopez@ansys.com, susannah.cooke@ansys.com.
[(1)] Ansys Iberia, Spain.
[(2)] Ansys UK, United Kingdom.

***Abstract*- En este artículo se propone el uso de la herramienta PyAEDT para llevar a cabo una sesión de laboratorio centrada en el análisis de antenas dipolo. Esta propuesta se fundamenta en la creciente integración de la programación en Python con las herramientas de Ansys, tanto en la industria como en la investigación. Esto ha sido posible gracias a las librerías PyAnsys, utilizadas para optimizar procesos y combinar resultados de simulación con el ecosistema Python. Debido al aumento en la utilización de estas librerías y a las oportunidades que ofrecen en el ámbito académico, tanto para reforzar el aprendizaje técnico a través de simulaciones y explorar soluciones multidisciplinarias como para conectar a los alumnos con habilidades requeridas como futuros ingenieros, se han comenzado una serie de iniciativas para introducir PyAnsys a alumnos de grados de ingeniería. Basándonos en los exitosos resultados obtenidos en eventos anteriores en otras disciplinas, proponemos en este artículo la introducción de PyAEDT a estudiantes de Telecomunicaciones mediante la simulación de una antena dipolo como parte de una sesión de laboratorio de antenas. Se detallará el flujo de instrucciones de código utilizado, junto con los resultados obtenidos. Concluimos destacando los posibles beneficios de esta propuesta y así como futuras líneas de trabajo basadas en el ejemplo presentado en este artículo.**

I. INTRODUCCIÓN

En base a la experiencia en sectores industriales y en las valoraciones de la comunidad académica que Ansys tiene, se ha detectado una brecha entre lo que la industria espera de estudiantes recién graduados y la formación real que éstos poseen al finalizar los estudios [1]. Dentro de esta brecha se encuentra la formación en simulación y el manejo de herramientas comerciales como Ansys para la resolución de problemas de ingeniería. En industrias como automoción, telecomunicaciones, obra civil, etc., la simulación es esencial durante los procesos de diseño para reducir tiempos y costes. Sin embargo, las empresas del sector coinciden en la escasa formación que en general los alumnos recién graduados poseen en el manejo de dichas herramientas de simulación. Por ello, parte del esfuerzo del equipo académico de Ansys se centra en el soporte y ayuda a profesores de universidades para introducir simulación en sus contenidos educativos, de forma que los alumnos empiecen a familiarizarse y a manejar estas herramientas desde cursos iniciales.

Además, en 2020, Ansys comienza a trabajar en la iniciativa de las librerías PyAnsys [2], [3] que posibilitan el manejo de los productos de Ansys mediante la programación en Python. Actualmente, se están desarrollando interfaces Python para las diferentes herramientas Ansys, por ejemplo, PyFluent (Python - Ansys Fluent), PyMAPDL (Python - Ansys MAPDL) y PyAEDT (Python - Ansys Electronics Desktop, AEDT), entre otras. La capacidad de interactuar con todas las herramientas de Ansys mediante programación marca el inicio de un nuevo paradigma para la industria, los investigadores y los estudiantes, generando oportunidades innovadoras de colaboración y resolución de problemas a través de un lenguaje común. La concepción de PyAnsys busca establecer un puente entre los *solvers* de Ansys y un extenso ecosistema representado por Python. De esta manera, se logra abordar problemas de ingeniería conectando los resultados de simulación con diversos elementos, como rutinas de optimización o algoritmos de aprendizaje automático (Machine Learning - ML), entre otros. Además, esta iniciativa concuerda con la tendencia actual de la ingeniería hacia simulaciones mutifísicas y multidisciplinares complejas basadas en el trabajo colaborativo y en equipo.

Hasta ahora, investigadores y empresas ya se están beneficiando de la posibilidad que PyAnsys ofrece (y en particular PyAEDT [4]-[6]), para dotar de mayor rapidez sus procesos de diseño, evitando la parte manual de realizar simulaciones repetitivas y aprovechando la posibilidad de optimización de resultados y automatización de simulaciones. No obstante, esta evolución también abre una ventana de oportunidad para expandir la formación de los estudiantes de ingeniería y acercarlos aún más al mundo de la simulación, sin necesidad de exigir un dominio avanzado de herramientas específicas, lo que resulta en una reducción significativa de la complejidad asociada a la simulación.

En base a esto, en este artículo se revisan las dos experiencias piloto que Ansys comenzó a poner en marcha el año pasado para introducir PyAnsys a alumnos de universidad [3] en la Sección II. A la vista de la buena aceptación, se propone en la Sección III la adaptación de una sesión de laboratorio de antenas usando PyAEDT. Se describe tanto el flujo de trabajo como el código necesario para la creación, análisis y obtención de resultados de una antena dipolo. Las conclusiones y las líneas futuras de este trabajo se presentan en la Sección IV.

II. EVENTOS PREVIOS

Teniendo en cuenta la importancia que tienen programación y simulación en carreras de ingeniería y las

posibilidades que combinar estas dos ofrecen, Ansys comienza la iniciativa de introducir PyAnsys a estudiantes universitarios, a través de dos tipos de experiencias piloto en colaboración con diferentes universidades [3].

A. *Eventos CodeFest*

El objetivo de estos eventos es que estudiantes utilicen PyAnsys para explorar diferentes posibilidades a la hora de abordar un problema de ingeniería, expandiendo así sus capacidades para resolver problemas a través de la programación. Se propone utilizar la librería PyMAPDL (Python + Ansys MAPDL) para resolver un problema sobre Mecánica. Se han celebrado un total de 4 eventos en las universidades de Cornell University (2 ediciones), Virginia Tech University y Karlsruhe Institute of Technology. Eventos que duraron uno o dos días, según la edición y contaron con la participación de 71, 84, 38 y 36 estudiantes, respectivamente. Estos eventos consistieron en resolver por grupos el problema propuesto, con premios para motivar la creatividad y otros aspectos relevantes del trabajo en equipo y la consecución de objetivos.

La idea inicialmente estaba destinada a alumnos de Grado, sin embargo, una gran parte de los asistentes fueron finalmente estudiantes de Máster y Doctorado. Una posible explicación es que los alumnos de últimos cursos podrían estar más enfocados en la aplicación y conexión de los conocimientos adquiridos, así como en completar su formación para sus futuros trabajos. Esta iniciativa fue ofertada para todas las carreras de ingeniería y contó con la participación de estudiantes principalmente de la rama de ingeniería mecánica e informática pero también estudiantes de ingeniería aeronáutica, electrónica y de materiales, entre otras disciplinas. La mayoría de los estudiantes contaban con una experiencia básica (nivel inicial e intermedio mayoritariamente) tanto en programación como en la utilización de las herramientas de simulación Ansys.

Sin embargo, los *feedback*s por parte de profesores y estudiantes fueron muy buenos, destacando los beneficios del trabajo en grupo y el aprendizaje de nuevas herramientas tecnológicas relevantes para su futuro. Cabe resaltar los resultados obtenidos por aquellos equipos formados por estudiantes de diferentes disciplinas, combinando así sus conocimientos técnicos y de programación para resolver los problemas propuestos. En general, la conclusión de los participantes en estos eventos es que más del 75% se ve utilizando PyAnsys en sus futuros trabajos como ingenieros.

B. *Sesiones 'Lab in a box'*

El objetivo de esta propuesta es posibilitar que estudiantes sin formación o experiencia previa en el manejo de la interfaz Ansys Fluent puedan beneficiarse de las ventajas que la simulación de problemas reales tiene en su aprendizaje. En otras palabras, ayudar a estudiantes de cursos iniciales de ingeniería a simular mediante la programación contenidos estudiados en clases teóricas, reduciendo la complejidad del manejo de la herramienta de simulación concreta. Esta idea surge de un caso práctico de laboratorio llevado a cabo cada año por algunos estudiantes de primeros cursos de ingeniería en Dinámica de Fluidos. La finalidad de la práctica consiste en replicar los resultados que se pueden obtener directamente usando la herramienta Ansys Fluent a través de la programación en Python usando PyFluent.

La prueba de concepto de esta sesión de laboratorio fue fácil de crear, lo cual demuestra que profesores y ayudantes de laboratorio interesados podrían customizar y adaptar fácilmente sus sesiones de laboratorio en este y otros campos al uso de Python.

La respuesta de los alumnos está siendo buena y tal y como se mencionaba en [3] como línea futura, se ha creado un recuso educativo con este contenido accesible para todos en el espacio educativo de Ansys [7].

C. *Resultados generales*

Tras estas experiencias piloto, se recogieron comentarios y valoraciones tanto de alumnos como de los profesores que participaron en estos eventos. En general, se observó gran interés por parte de ambos grupos en la combinación de la programación con herramientas de simulación. Tanto alumnos como los profesores confirmaron que estas experiencias habían contribuido a mejorar su formación, consolidando mediante simulación sus resultados de aprendizaje. Además, concluyeron en que PyAnsys les resultaba una herramienta muy útil para la resolución de casos simples o incluso problemas más complejos de ingeniería [3].

III. PyAEDT: Ejercicio práctico de simulación de una antena dipolo

En base a las experiencias previas utilizando PyMAPDL y PyFluent descritas en la Sección II, el objetivo de este artículo es continuar la iniciativa de introducir PyAnsys a estudiantes de ingeniería, en este caso, en el área de electromagnetismo. Para ello, se presenta en esta sección una propuesta para una sesión de laboratorio de antenas (destinada a alumnos de tercer o cuarto curso de Telecomunicaciones), combinando simulación y programación mediante el uso de PyAEDT.

A. *Motivación*

PyAEDT (Python + Ansys EDT) [2] ya se utiliza en la industria para la optimización y automatización en procesos de diseño, para el lanzamiento de simulaciones repetitivas y para combinar resultados de Ansys EDT con ML y otros recursos disponibles en el ecosistema Python. Por ejemplo, podemos destacar el uso de PyAEDT para automatizar el proceso de simulación en Ansys HFSS necesario para la colocación de antenas [4]. Además, las librerías PyAEDT ya están siendo una herramienta muy útil en el campo de la investigación. Encontramos ejemplos de simulación con PyAEDT en campos como antenas de lente *Luneburg* [5] o *THz phased array* [6], entre otros.

Debido a este creciente uso y con el objetivo de introducir esta herramienta a estudiantes de ingeniera como parte de su formación, en esta sección presentamos un caso sencillo para que éstos puedan ir adentrándose en el mundo PyAEDT.

B. *Propuesta de sesión de laboratorio: Simulación de una antena dipolo usando PyAEDT*

La práctica que se plantea en este artículo se basa en un ejemplo sencillo que se encuentra documentado, entre otros, en la web de PyAnsys [2]. Ahí también podremos encontrar otra información útil y complementaria sobre cómo instalar y empezar con PyAEDT (*user guide* – recursos que podrían complementar esta sesión de laboratorio en el caso de que los estudiantes no tengan conocimiento previo de Python o de las

instrucciones específicas de PyAEDT). Además, en dicha web están disponibles más ejemplos que podrían aplicarse a otros campos de la enseñanza del electromagnetismo, como la simulación de un filtro inductivo en guía de onda usando Ansys HFSS, la implementación de un circuito LRC en Ansys Circuit o ejemplos multidisciplinares como el análisis de un cable coaxial combinando la simulación electromagnética usando HFSS y térmica usando Icepak.

Dentro de los ejemplos disponibles, en este artículo nos centraremos en el análisis de una antena dipolo - un contenido habitual en el programa de antenas en grados de Telecomunicación. De esta forma, profesores podrían beneficiarse del código presentado para poder combinar programación y simulación en algunas de sus sesiones de laboratorio.

Para la definición y análisis de un dipolo usando PyAEDT, el flujo de trabajo será el mostrado en la Fig. 1. Los pasos a seguir coinciden con los necesarios para llevar a cabo el mismo análisis en Ansys HFSS. La diferencia radica en que, en este caso, en lugar completar esta secuencia de acciones manualmente manejando directamente las opciones de la interfaz de HFSS, las realizamos con los comandos sugeridos en el Apéndice V.

Cabe destacar aquí la flexibilidad en cuanto a las diferentes opciones de proporcionar estos comandos a los alumnos, pudiendo escoger si estos comandos son directamente facilitados, guiándolos en áreas clave que ellos puedan modificar o extender; entregados a los estudiantes de forma parcialmente completa, fomentando que ellos busquen e implementen funcionalidades específicas; o completamente ocultos detrás de otra interfaz de usuario (como un cuaderno Jupyter). Esta elección dependerá de los objetivos de aprendizaje que se pretendan abordar y el nivel previo (en simulación y programación) del conjunto específico de alumnos.

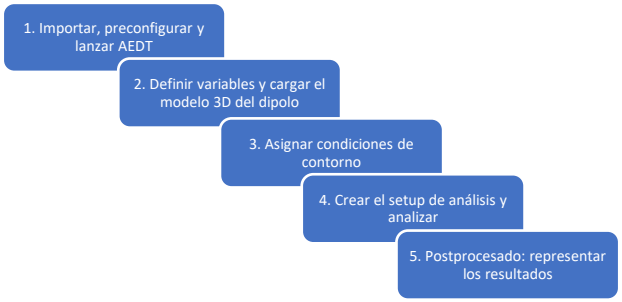

Fig. 1. *Workflow* del código Python del Apéndice V para crear y analizar una antena dipolo.

Es importante mencionar que el *solver* utilizado para resolver el problema mediante las instrucciones PyAEDT es el mismo que el de Ansys HFSS, el Método de Elementos Finitos (FEM). En las subsecciones del Apéndice V, encontramos los comandos relativos al *workflow* de la Fig. 1. Para importar, preconfigurar y enlazar con el programa AEDT (paso 1) ejecutamos los comandos de la subsección V.A. Las siguientes tareas (paso 2) serían la definición de las variables y creación del modelo deseado (en este ejemplo el modelo de antena dipolo con su longitud definida como una variable), mediante el código de la subsección V.B. En este caso, la excitación del problema como *Lumped Port* ya viene incluida en el modelo del dipolo. Ejecutando los comandos de la subsección V.C. creamos las condiciones de contorno para posteriormente definir nuestro *setup* de análisis y analizar el modelo utilizando las instrucciones de la subsección V.D. (pasos 3 y 4). Finalmente, podemos visualizar los resultados del análisis creando dos gráficas, una para el parámetro S11 en dB y otra para visualizar el diagrama de radiación, mediante el código de la subsección V.E. Siguiendo esta secuencia de instrucciones obtenemos el modelo de HFSS mostrado en la Fig. 2, destacando que mediante este *workflow* hemos sido capaces de crear tanto la geometría como las condiciones de contorno, la excitación y el barrido en frecuencia deseado. Los resultados de análisis obtenidos mediante este código se muestran en la Fig. 3.

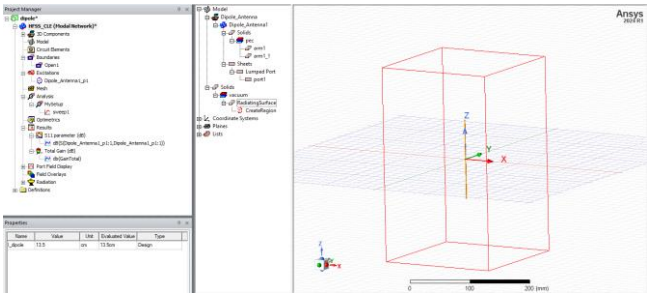

Fig. 2. Modelo, condiciones de contorno, excitación y *setup* para el análisis de una antena dipolo mediante el código presentado en el Apéndice V.

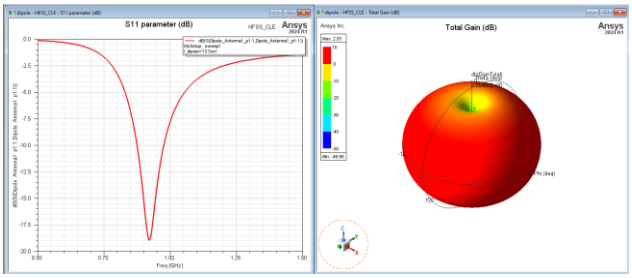

Fig. 3. Parámetro S11 y ganancia total de la antena de la Fig. 2, obtenidos mediante el código presentado en el Apéndice V.

El tiempo de ejecución es de 20 segundos, consumiendo 220 MB de RAM en un ordenador i7, usando 4 cores.

Un posible caso de exploración para los alumnos sería cambiar la longitud del dipolo para centrar el funcionamiento del dipolo en 1 GHz. Para esto, los alumnos pueden tener dos opciones. La primera sería disminuir la longitud del dipolo y volver a simular hasta conseguir centrar el mínimo del parámetro S11 en 1 GHz. Ejecutando las instrucciones recogidas en la subsección V.F. obtenemos los resultados mostrados en la Fig. 4.

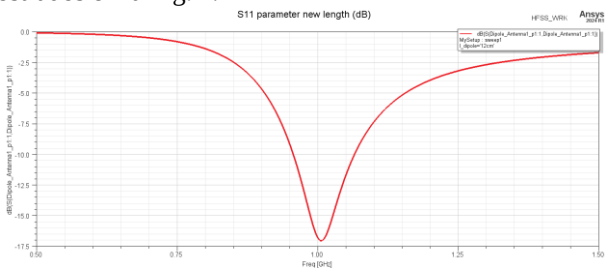

Fig. 4. Parámetro S11 de la antena de la Fig. 2, con una longitud del dipolo igual a 12 cm, obtenidos mediante el código presentado en la subsección V.F.

La otra posibilidad es realizar un análisis paramétrico variando la longitud del dipolo para posteriormente seleccionar la más adecuada. Esto lo realizamos siguiendo las instrucciones presentadas en el Apéndice V.G. Los resultados obtenidos con este código se muestran en la Fig. 5. En este

caso, la propuesta es que los alumnos sean capaces de seleccionar la longitud de dipolo más adecuada según la frecuencia deseada de funcionamiento de la antena.

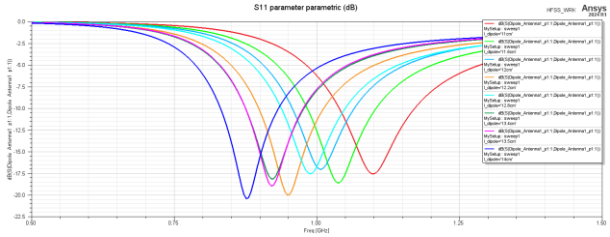

Fig. 5. Resultados (parámetro S11) del análisis paramétrico variando la longitud del dipolo mediante el código presentado en la subsección V.G.

Una vez finalizada la práctica, el entorno AEDT puede cerrarse mediante la instrucción de la subsección V.H.

IV. Conclusiones y líneas futuras

En este artículo se propone una sencilla práctica de laboratorio que consiste en el análisis una antena dipolo en Ansys HFSS usando Python. La motivación de este trabajo reside en las ventajas que aprender a utilizar programación con un software industrial puede tener para los alumnos en su futuro profesional y en los buenos *feedbacks* recibidos previamente de alumnos y profesores en el uso de PyAnsys para la solución de problemas de ingeniería en otras disciplinas. En base a esto, se han revisado experiencias piloto previas llevadas a cabo en diferentes universidades, para luego presentar un caso práctico de uso de PyAEDT en una sesión de laboratorio de antenas. Se ha presentado tanto el *workflow* como las instrucciones específicas en Python, así como los resultados obtenidos y posibles extensiones para que los alumnos exploren diferentes opciones de diseño.

Como línea futura, se pretende desarrollar un recurso educativo accesible para todos los profesores de antenas interesados en complementar sus sesiones de laboratorio con éste y otros ejemplos sencillos, como una cosimulación con Ansys Circuit para mejorar la adaptación de la antena, usando PyAEDT.

V. Apéndice

Código para la creación y análisis de una antena dipolo en Ansys HFSS usando Python (PyAEDT versión 0.7.12 con AEDT 2024R1):

A. Inicialización

```
    import os
    import pyaedt
    project_name =
pyaedt.generate_unique_project_name(project_name="dipole")
    non_graphical = False
    d = pyaedt.launch_desktop("2024.1",
non_graphical=non_graphical, new_desktop_session=True)
    hfss = pyaedt.Hfss(projectname=project_name,
solution_type="Modal")
```

B. Creación de variables y componente (dipolo)

```
    hfss["l_dipole"] = "13.5cm"
    compfile = hfss.components3d["Dipole_Antenna_DM"]
    geometryparams =
hfss.get_components3d_vars("Dipole_Antenna_DM")
    geometryparams["dipole_length"] = "l_dipole"
    hfss.modeler.insert_3d_component(compfile,
geometryparams)
```

C. Condiciones de contorno

```
    hfss.create_open_region(Frequency="1GHz")
```

D. Definición del setup y análisis

```
    setup = hfss.create_setup("MySetup")
    setup.props["Frequency"] = "1GHz"
    setup.props["MaximumPasses"] = 1
    hfss.create_linear_count_sweep(
        setupname=setup.name,
        unit="GHz",
        freqstart=0.5,
        freqstop=1.5,
        num_of_freq_points=251,
        sweepname="sweep1",
        sweep_type="Interpolating",
         interpolation_tol=3,
        interpolation_max_solutions=255,
        save_fields=False,)
    hfss.analyze_setup("MySetup")
```

E. Representar resultados

```
    hfss.create_scattering("S11 parameter (dB)")
    variations =
hfss.available_variations.nominal_w_values_dict
    variations["Freq"] = ["1GHz"]
    variations["Theta"] = ["All"]
    variations["Phi"] = ["All"]
    new_report =
hfss.post.reports_by_category.far_field("db(GainTotal)",
hfss.nominal_adaptive, "3D")
    new_report.variations = variations
    new_report.report_type = "3D Polar Plot"
    new_report.secondary_sweep = "Theta"
    new_report.create("Total Gain (dB)")
```

Código extendido para modificar la longitud del dipolo y ejecutar un análisis paramétrico usando la consola de Python:

F. Modificar la longitud del dipolo

```
    hfss["l_dipole"] = "12cm"
    hfss.analyze_setup("MySetup")
    hfss.create_scattering("S11 parameter new length (dB)")
```

G. Análisis paramétrico

```
    sweep = hfss.parametrics.add("l_dipole", 11, 14, 6)
    sweep.analyze()
    nw=hfss.create_scattering("S11 parameter parametric
(dB)")
    variations["l_dipole"] = ["All"]
    variations["Freq"] = ["All"]
    nw.variations = variations
    nw.create("S11 parameter parametric (dB)")
```

H. Cerrar AEDT

```
    d.release_desktop()
```

Referencias

[1] K. Alboaouh, "The Gap Between Engineering Schools and Industry: A Strategic Initiative," *2018 IEEE Frontiers in Education Conference (FIE)*, San Jose, CA, USA, 2018, pp. 1-6, doi: 10.1109/FIE.2018.8659314.

[2] *https://docs.pyansys.com/version/dev/*

[3] S. C. Cooke, S. Coleman, J. Derrick, "Exploring the potential for scripting with simulation in engineering education – practical examples using Python and Ansys," *European Society for Engineering Education (SEFI)*. DOI: 10.21427/YQXN-G372.

[4] *https://www.ansys.com/webinars/automating-hfss-antenna-placement-workflow-with-pyaedt*

[5] S. Biswas, "Design & EM Simulation of a Continuous-Dielectric Luneburg Lens Antenna Using COMSOL-PyAEDT," *2023 IEEE International Symposium on Antennas and Propagation and USNC-URSI Radio Science Meeting* (USNC-URSI), Portland, OR, USA, 2023, pp. 639-640, doi: 10.1109/USNC-URSI52151.2023.10238151.

[6] L. -Y. O. Yang, W. -H. Wu and B. Tsai, "EIRP Optimization of a Phased Array With Mutual Coupling for THz Applications," *2022 IEEE Conference on Antenna Measurements and Applications (CAMA)*, Guangzhou, China, 2022, pp. 1-3, doi: 10.1109/CAMA56352.2022.10002699.

[7] *https://www.ansys.com/academic/educators/education-resources/flow-around-a-cylinder-with-ansys-fluent-and-jupyter-lab*