A rational mechanics course where everything is made with Python code

Bettachini, Víctor A.; Real, Mariano A.; Palazzo, Edgardo

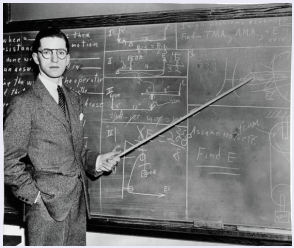
New Media Pedagogy 23

Value students and professors time

Licklider (1957): 85 % of “thinking” are actually mundane task (calculations, drawing, etc.)

Value students and professors time

Licklider (1957): 85 % of "thinking" are actually mundane task (calculations, drawing, etc.)

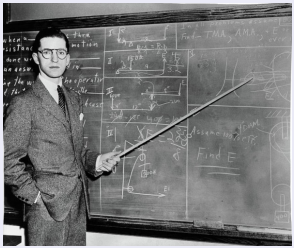


Classroom and practice: an excercise on transcription

- Professor: lessons $\xrightarrow{\textit{by heart}}$ blackboard/slides

Value students and professors time

Licklider (1957): 85 % of "thinking" are actually mundane task (calculations, drawing, etc.)

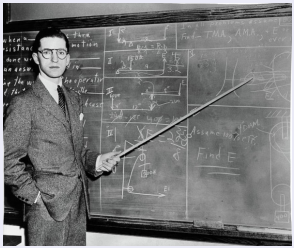


Classroom and practice: an excercise on transcription

- Professor: lessons $\xrightarrow{\textit{by heart}}$ blackboard/slides
- Student: blackboard/slides $\xrightarrow{\textit{copies}}$ notebooks

Value students and professors time

Licklider (1957): 85 % of "thinking" are actually mundane task (calculations, drawing, etc.)

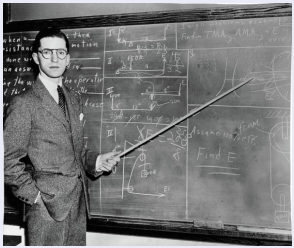


Classroom and practice: an excercise on transcription

- Professor: lessons $\xrightarrow{\textit{by heart}}$ blackboard/slides
- Student: blackboard/slides $\xrightarrow{\textit{copies}}$ notebooks
- Práctice **reiterate** diagrames, calculations, etc.

Value students and professors time

Licklider (1957): 85 % of "thinking" are actually mundane task (calculations, drawing, etc.)

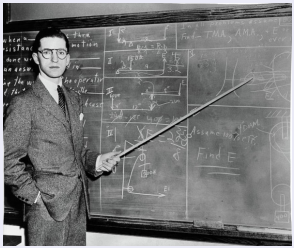


Classroom and practice: an excercise on transcription

- Professor: lessons $\xrightarrow{\textit{by heart}}$ blackboard/slides
- Student: blackboard/slides $\xrightarrow{\textit{copies}}$ notebooks
- Práctice **reiterate** diagrames, calculations, etc.
- Boredom $\Longrightarrow \downarrow$ concentration on the subject

Value students and professors time

Licklider (1957): 85 % of "thinking" are actually mundane task (calculations, drawing, etc.)

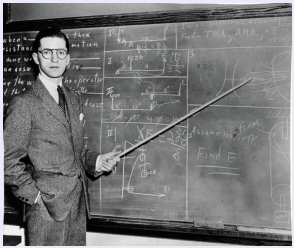


Classroom and practice: an excercise on transcription

- Professor: lessons $\xrightarrow{\textit{by heart}}$ blackboard/slides
- Student: blackboard/slides $\xrightarrow{\textit{copies}}$ notebooks
- Práctice **reiterate** diagrames, calculations, etc.
- Boredom $\Longrightarrow \downarrow$ concentration on the subject

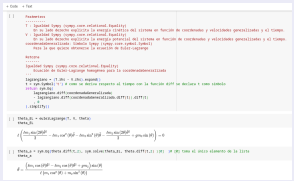
Value students and professors time

Licklider (1957): 85 % of "thinking" are actually mundane task (calculations, drawing, etc.)



Classroom and practice: an excercise on transcription

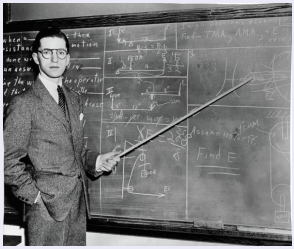
- Professor: lessons $\xrightarrow{\textit{by heart}}$ blackboard/slides
- Student: blackboard/slides $\xrightarrow{\textit{copies}}$ notebooks
- Práctice **reiterate** diagrames, calculations, etc.
- Boredom $\implies \downarrow$ concentration on the subject



- Professor: ideas $\rightarrow$ new code/notes in repository

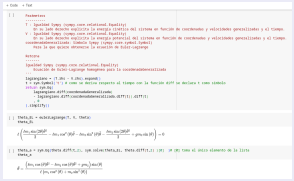
Value students and professors time

Licklider (1957): 85 % of "thinking" are actually mundane task (calculations, drawing, etc.)



Classroom and practice: an excercise on transcription

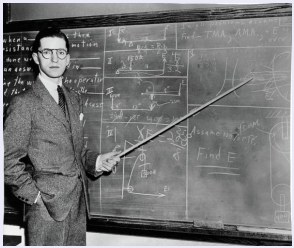
- Professor: lessons $\xrightarrow{\textit{by heart}}$ blackboard/slides
- Student: blackboard/slides $\xrightarrow{\textit{copies}}$ notebooks
- Práctice **reiterate** diagrames, calculations, etc.
- Boredom $\Longrightarrow \downarrow$ concentration on the subject



- Professor: ideas $\rightarrow$ new code/notes in repository
- Student: course repository $\rightarrow$ its own modifiable one

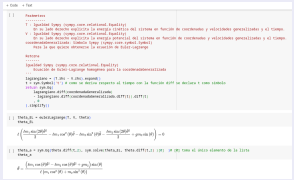
Value students and professors time

Licklider (1957): 85 % of "thinking" are actually mundane task (calculations, drawing, etc.)



Classroom and practice: an excercise on transcription

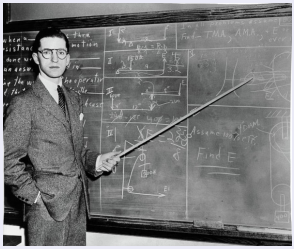
- Professor: lessons $\xrightarrow{\textit{by heart}}$ blackboard/slides
- Student: blackboard/slides $\xrightarrow{\textit{copies}}$ notebooks
- Práctice **reiterate** diagrames, calculations, etc.
- Boredom $\implies \downarrow$ concentration on the subject



- Professor: ideas $\rightarrow$ new code/notes in repository
- Student: course repository $\rightarrow$ its own modifiable one
- Use code to solve problems = **recycle** professor's code

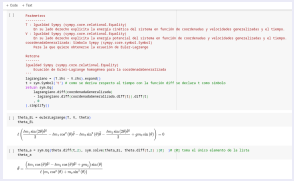
Value students and professors time

Licklider (1957): 85 % of "thinking" are actually mundane task (calculations, drawing, etc.)



Classroom and practice: an excercise on transcription

- Professor: lessons $\xrightarrow{\textit{by heart}}$ blackboard/slides
- Student: blackboard/slides $\xrightarrow{\textit{copies}}$ notebooks
- Práctice **reiterate** diagrames, calculations, etc.
- Boredom $\implies \downarrow$ concentration on the subject



- Professor: ideas $\rightarrow$ new code/notes in repository
- Student: course repository $\rightarrow$ its own modifiable one
- Use code to solve problems = **recycle** professor's code
- Modifiying it solves different problems

Engineering students must take advantage of code at every single lecture

Engineering students must take advantage of code at every single lecture

- Currently they use a pocket calculator **after they learnt** learning arithmetics at school

Engineering students must take advantage of code at every single lecture

- Currently they use a pocket calculator **after they learnt** learning arithmetics at school
- They'll employ computational algebra **after they learnt** algebra and calculus

[14]:
```
sistemaEcuaciones = [
    x_EL,
    phi_EL,
]
variablesDespeje = [x.diff(t,2), phi.diff(t,2)] # despejar aceleraciones generalizadas
variablesDespeje_sol= sym.nonlinsolve(sistemaEcuaciones, variablesDespeje ).args[0]
```

[15]:
```
x_pp = sym.Eq(variablesDespeje[0], variablesDespeje_sol.args[0] ) # [m s-2]
phi_pp = sym.Eq(variablesDespeje[1], variablesDespeje_sol.args[1] ) # [m s-2]
x_pp, phi_pp
```

[15]:
$$\left(\ddot{x} = \frac{-\ell g m_2 \sin(\phi) + \frac{\ell m_2 \left(\ell m_2 \cos(\phi)\dot{\phi}^2 + g m_1 + g m_2\right) \sin(\phi)}{m_1 + m_2 \sin^2(\phi)}}{\ell m_2 \cos(\phi)}, \quad \ddot{\phi} = -\frac{\left(\ell m_2 \cos(\phi)\dot{\phi}^2 + g m_1 + g m_2\right) \sin(\phi)}{\ell \left(m_1 + m_2 \sin^2(\phi)\right)} \right)$$

Engineering students must take advantage of code at every single lecture

- Currently they use a pocket calculator **after they learnt** learning arithmetics at school
- They'll employ computational algebra **after they learnt** algebra and calculus
  - Focus on new skills, not in automatable calculations

[14]:
```
sistemaEcuaciones = [
    x_EL,
    phi_EL,
]
variablesDespeje = [x.diff(t,2), phi.diff(t,2)] # despejar aceleraciones generalizadas
variablesDespeje_sol= sym.nonlinsolve(sistemaEcuaciones, variablesDespeje ).args[0]
```

[15]:
```
x_pp = sym.Eq(variablesDespeje[0], variablesDespeje_sol.args[0] ) # [m s-2]
phi_pp = sym.Eq(variablesDespeje[1], variablesDespeje_sol.args[1] ) # [m s-2]
x_pp, phi_pp
```

[15]:
$$\left(\ddot{x} = \frac{-\ell g m_2 \sin(\phi) + \frac{\ell m_2 \left(\ell m_2 \cos(\phi)\dot{\phi}^2 + g m_1 + g m_2\right) \sin(\phi)}{m_1 + m_2 \sin^2(\phi)}}{\ell m_2 \cos(\phi)}, \quad \ddot{\phi} = -\frac{\left(\ell m_2 \cos(\phi)\dot{\phi}^2 + g m_1 + g m_2\right) \sin(\phi)}{\ell \left(m_1 + m_2 \sin^2(\phi)\right)} \right)$$

Engineering students must take advantage of code at every single lecture

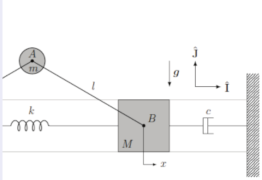
- Currently they use a pocket calculator **after they learnt** learning arithmetics at school
- They'll employ computational algebra **after they learnt** algebra and calculus
  - Focus on new skills, not in automatable calculations
  - Employing numerical calculus they solve what is impossible in a blackboard/paper

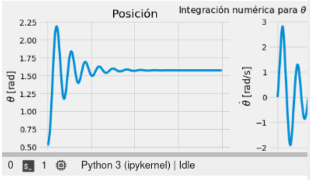
```
[14]: sistemaEcuaciones = [
          x_EL,
          phi_EL,
      ]
      variablesDespeje = [x.diff(t,2), phi.diff(t,2)] # despejar aceleraciones generalizadas
      variablesDespeje_sol= sym.nonlinsolve(sistemaEcuaciones, variablesDespeje ).args[0]

[15]: x_pp = sym.Eq(variablesDespeje[0], variablesDespeje_sol.args[0] ) # [m s-2]
      phi_pp = sym.Eq(variablesDespeje[1], variablesDespeje_sol.args[1] ) # [m s-2]
      x_pp, phi_pp
```

[15]:

$$\left(\ddot{x} = \frac{-\ell g m_2 \sin(\phi) + \frac{\ell m_2 \left(\ell m_2 \cos(\phi)\dot{\phi}^2 + g m_1 + g m_2\right)\sin(\phi)}{m_1 + m_2 \sin^2(\phi)}}{\ell m_2 \cos(\phi)}, \quad \ddot{\phi} = -\frac{\left(\ell m_2 \cos(\phi)\dot{\phi}^2 + g m_1 + g m_2\right)\sin(\phi)}{\ell\left(m_1 + m_2 \sin^2(\phi)\right)} \right)$$





Engineering students must take advantage of code at every single lecture

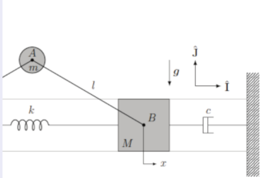
- Currently they use a pocket calculator **after they learnt** learning arithmetics at school
- They'll employ computational algebra **after they learnt** algebra and calculus
  - Focus on new skills, not in automatable calculations
  - Employing numerical calculus they solve what is impossible in a blackboard/paper

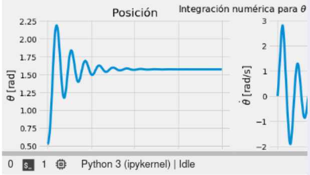
```
[14]: sistemaEcuaciones = [
          x_EL,
          phi_EL,
      ]
      variablesDespeje = [x.diff(t,2), phi.diff(t,2)] # despejar aceleraciones generalizadas
      variablesDespeje_sol= sym.nonlinsolve(sistemaEcuaciones, variablesDespeje ).args[0]
```

```
[15]: x_pp = sym.Eq(variablesDespeje[0], variablesDespeje_sol.args[0] ) # [m s-2]
      phi_pp = sym.Eq(variablesDespeje[1], variablesDespeje_sol.args[1] ) # [m s-2]
      x_pp, phi_pp
```

[15]:

$$\left(\ddot{x} = \frac{-\ell g m_2 \sin(\phi) + \frac{\ell m_2 \left(\ell m_2 \cos(\phi)\dot{\phi}^2 + g m_1 + g m_2\right)\sin(\phi)}{m_1 + m_2 \sin^2(\phi)}}{\ell m_2 \cos(\phi)}, \; \ddot{\phi} = -\frac{\left(\ell m_2 \cos(\phi)\dot{\phi}^2 + g m_1 + g m_2\right)\sin(\phi)}{\ell\left(m_1 + m_2 \sin^2(\phi)\right)} \right)$$





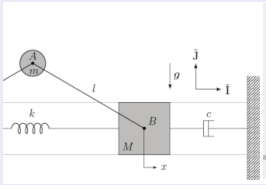
Engineering students must take advantage of code at every single lecture

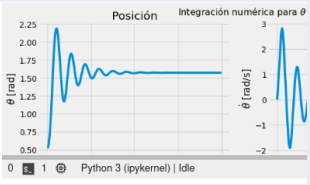
- Currently they use a pocket calculator **after they learnt** learning arithmetics at school
- They'll employ computational algebra **after they learnt** algebra and calculus
  - Focus on new skills, not in automatable calculations
  - Employing numerical calculus they solve what is impossible in a blackboard/paper

```
[14]: sistemaEcuaciones = [
          x_EL,
          phi_EL,
      ]
      variablesDespeje = [x.diff(t,2), phi.diff(t,2)] # despejar aceleraciones generalizadas
      variablesDespeje_sol= sym.nonlinsolve(sistemaEcuaciones, variablesDespeje ).args[0]

[15]: x_pp = sym.Eq(variablesDespeje[0], variablesDespeje_sol.args[0] ) # [m s-2]
      phi_pp = sym.Eq(variablesDespeje[1], variablesDespeje_sol.args[1] ) # [m s-2]
      x_pp, phi_pp

[15]: ( ẍ = (−ℓ g m2 sin(ϕ) + (ℓ m2 (ℓ m2 cos(ϕ) ϕ̇² + g m1 + g m2) sin(ϕ)) / (m1 + m2 sin²(ϕ))) / (ℓ m2 cos(ϕ)),
        ϕ̈ = −(ℓ m2 cos(ϕ) ϕ̇² + g m1 + g m2) sin(ϕ) / (ℓ (m1 + m2 sin²(ϕ))) )
```





Papert (1980) "...the best learning takes place when the learner takes charge"

- An expample problem is solved by the professor provided code

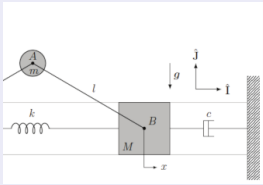
Engineering students must take advantage of code at every single lecture

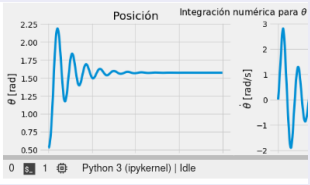
- Currently they use a pocket calculator **after they learnt** learning arithmetics at school
- They'll employ computational algebra **after they learnt** algebra and calculus
  - Focus on new skills, not in automatable calculations
  - Employing numerical calculus they solve what is impossible in a blackboard/paper

```
[14]: sistemaEcuaciones = [
          x_EL,
          phi_EL,
      ]
      variablesDespeje = [x.diff(t,2), phi.diff(t,2)] # despejar aceleraciones generalizadas
      variablesDespeje_sol= sym.nonlinsolve(sistemaEcuaciones, variablesDespeje ).args[0]

[15]: x_pp = sym.Eq(variablesDespeje[0], variablesDespeje_sol.args[0] ) # [m s-2]
      phi_pp = sym.Eq(variablesDespeje[1], variablesDespeje_sol.args[1] ) # [m s-2]
      x_pp, phi_pp

[15]: ( ẍ = (−ℓ g m2 sin(ϕ) + (ℓ m2 (ℓ m2 cos(ϕ) ϕ̇² + g m1 + g m2) sin(ϕ)) / (m1 + m2 sin²(ϕ))) / (ℓ m2 cos(ϕ)),
        ϕ̈ = −(ℓ m2 cos(ϕ) ϕ̇² + g m1 + g m2) sin(ϕ) / (ℓ (m1 + m2 sin²(ϕ))) )
```





Papert (1980) "...the best learning takes place when the learner takes charge"

- An expample problem is solved by the professor provided code
- The student modifies it to solve other related problems

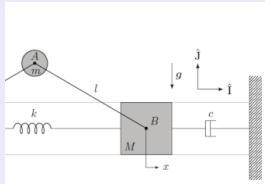
Engineering students must take advantage of code at every single lecture

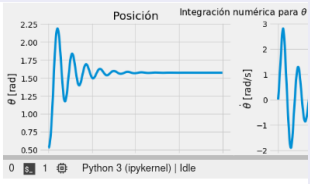
- Currently they use a pocket calculator **after they learnt** learning arithmetics at school
- They'll employ computational algebra **after they learnt** algebra and calculus
  - Focus on new skills, not in automatable calculations
  - Employing numerical calculus they solve what is impossible in a blackboard/paper

```
[14]: sistemaEcuaciones = [
          x_EL,
          phi_EL,
      ]
      variablesDespeje = [x.diff(t,2), phi.diff(t,2)] # despejar aceleraciones generalizadas
      variablesDespeje_sol= sym.nonlinsolve(sistemaEcuaciones, variablesDespeje ).args[0]

[15]: x_pp = sym.Eq(variablesDespeje[0], variablesDespeje_sol.args[0] ) # [m s-2]
      phi_pp = sym.Eq(variablesDespeje[1], variablesDespeje_sol.args[1] ) # [m s-2]
      x_pp, phi_pp

[15]: ⎛ẍ = (−ℓ g m2 sin(ϕ) + (ℓ m2 (ℓ m2 cos(ϕ) ϕ̇² + g m1 + g m2) sin(ϕ)) / (m1 + m2 sin²(ϕ))) / (ℓ m2 cos(ϕ)), ϕ̈ = −(ℓ m2 cos(ϕ) ϕ̇² + g m1 + g m2) sin(ϕ) / (ℓ (m1 + m2 sin²(ϕ)))⎞
```





Papert (1980) "...the best learning takes place when the learner takes charge"

- An expample problem is solved by the professor provided code
- The student modifies it to solve other related problems
- Gradually he becomes autonomous by reusing not the provided but his own code

All course material can be edited on-line

All course material can be edited on-line

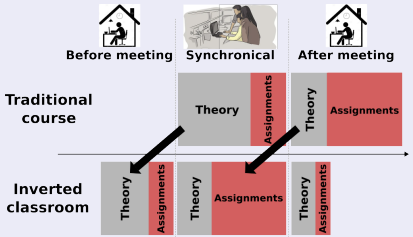
On-line programmable notebook: text + equations + code

Synchronical and asynchronical work on the code

New theory alongisde its worked examples in programmable notebooks

- On-line 24/7 **asynchronical** consultations that are **public** for others to see

Inverted classroom



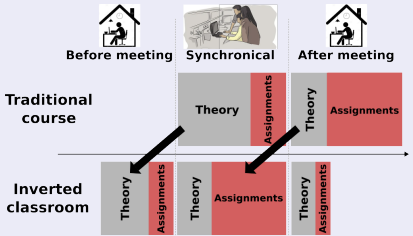
Synchronical	Theory	Assignments
Before	Read and apply	Start them
During	Consultations	Complete them
After	Additional consultations	TA's corrections

Synchronical and asynchronical work on the code

New theory alongisde its worked examples in programmable notebooks

- On-line 24/7 **asynchronical** consultations that are **public** for others to see
- **Remote collaboration** on multi-user notebooks

Inverted classroom



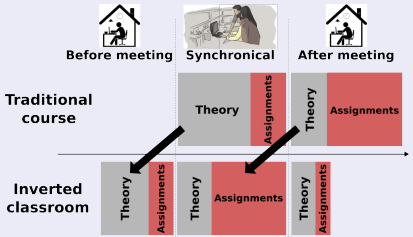
Synchronical	Theory	Assignments
Before	Read and apply	Start them
During	Consultations	Complete them
After	Additional consultations	TA's corrections

Synchronical and asynchronical work on the code

New theory alongisde its worked examples in programmable notebooks

- On-line 24/7 **asynchronical** consultations that are **public** for others to see
- **Remote collaboration** on multi-user notebooks
- Weekly meetings to **synchronically** unfinished assignments with TA's assistance

Inverted classroom



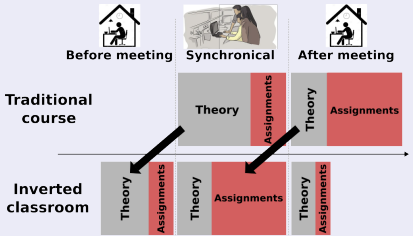
Synchronical	Theory	Assignments
Before	Read and apply	Start them
During	Consultations	Complete them
After	Additional consultations	TA's corrections

Synchronical and asynchronical work on the code

New theory alongisde its worked examples in programmable notebooks

- On-line 24/7 **asynchronical** consultations that are **public** for others to see
- **Remote collaboration** on multi-user notebooks
- Weekly meetings to **synchronically** unfinished assignments with TA's assistance
- On a weekly basis these **must** be turned-in for scoring

Inverted classroom



Synchronical	Theory	Assignments
Before	Read and apply	Start them
During	Consultations	Complete them
After	Additional consultations	TA's corrections

Asynchronical corrections and remote assistance

Student's work can be commented and edited in Google Colaboratory

07 No conservativas | ej4

Archivo Editar Ver Insertar Entorno de ejecución Herramientas Ayuda Se editó por última vez: 3 de junio

Comentar Compartir

+ Código + Texto

Conectar Editando

```
[ ] # Energía potencial
    m1_V = - (m1* g* (- N.y)).dot(m1_r)
    # pot_k1 = unMedio* ( -k1* ((l10 + x1)* (sym.cos(theta) - sym.sin(theta)) )**2 ) # mal
    pot_k1 = unMedio* k1* (l10 + x1)**2 # Lo escribí yo
    # pot_k2 = unMedio* -k2* (l20 + x)**2
    pot_k2 = unMedio* k2* (l20 + x)**2
    V = sym.Eq(sym.Symbol('V'), m1_V + pot_k1 + pot_k2 ) #agrega el potencial elastico k en la ecuacion
    V
```

$$V = gm_1\left(-l_{10} - x_1\right)\sin\left(\theta\right) + \frac{k_1\left(l_{10} + x_1\right)^2}{2} + \frac{k_2\left(l_{20} + x\right)^2}{2}$$

Victor Alexis Bettachini Resolver
31 de may. de 2021
(editado el 31 de may. de 2021)

- El estiramiento del resorte de k_1 es colineal con x1. No tienen sentido pensar en proyecciones (si es lo que hiciste, que realmente no entiendo).
- ¿Porque negativos los k?

Lagrangiano

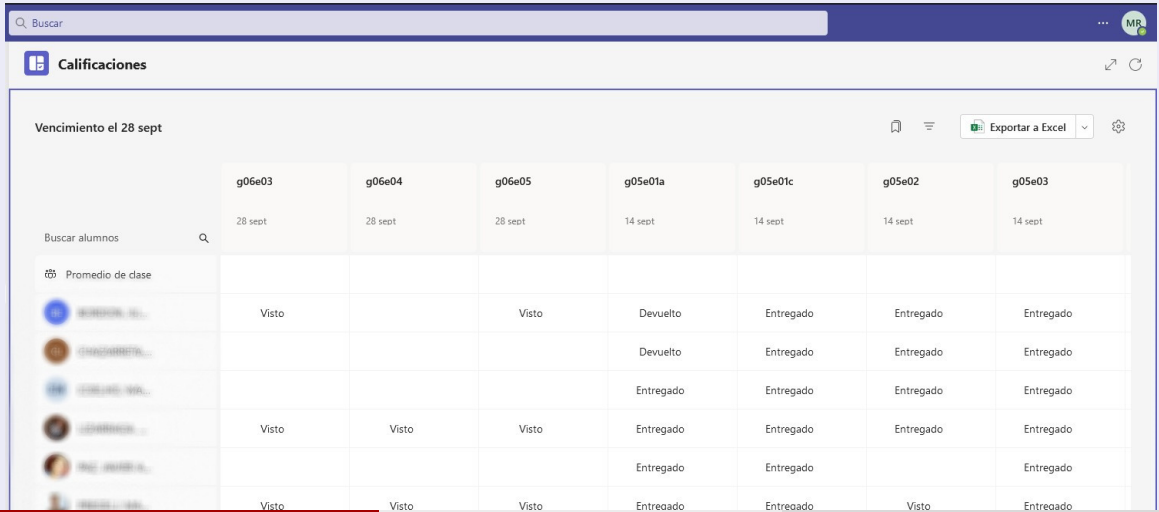
```
[ ] L = sym.Eq(sym.Symbol('\mathcal{L}'), (T.rhs - V.rhs))
    L
```

$$\mathcal{L} = -gm_1\left(-l_{10} - x_1\right)\sin\left(\theta\right) - \frac{k_1\left(l_{10} + x_1\right)^2}{2} - \frac{k_2\left(l_{20} + x\right)^2}{2} + \frac{\left(m_0 + m_1\right)\left(2\cos\left(\theta\right)\dot{x}\dot{x}_1 + \dot{x}^2 + \dot{x}_1^2\right)}{2}$$

ECUACIONES DE EULER

Individualized student follow-up at Microsoft Teams

A record of the weekly turn-up of assignments



Summary

A course centred on code

- Theory: text + equations + executable cod in digital notebooks.

Inverted classroom

Summary

A course centred on code

- Theory: text + equations + executable cod in digital notebooks.
- Reinforced by: suggested bibliography and short professor's videos.

Inverted classroom

Summary

A course centred on code

- Theory: text + equations + executable cod in digital notebooks.
- Reinforced by: suggested bibliography and short professor's videos.
- assignments: professor's code recycling.

Inverted classroom

Summary

A course centred on code

- Theory: text + equations + executable cod in digital notebooks.
- Reinforced by: suggested bibliography and short professor's videos.
- assignments: professor's code recycling.
- On-line:

Inverted classroom

Summary

A course centred on code

- Theory: text + equations + executable cod in digital notebooks.
- Reinforced by: suggested bibliography and short professor's videos.
- assignments: professor's code recycling.
- On-line:
  - Remote collaboration and correction

Inverted classroom

Summary

A course centred on code

- Theory: text + equations + executable cod in digital notebooks.
- Reinforced by: suggested bibliography and short professor's videos.
- assignments: professor's code recycling.
- On-line:
  - Remote collaboration and correction
  - Doesn't require powerful nor at campus computers

Inverted classroom

Summary

A course centred on code

- Theory: text + equations + executable cod in digital notebooks.
- Reinforced by: suggested bibliography and short professor's videos.
- assignments: professor's code recycling.
- On-line:
  - Remote collaboration and correction
  - Doesn't require powerful nor at campus computers
  - A dated record of each student's work

Inverted classroom

Summary

A course centred on code

- Theory: text + equations + executable cod in digital notebooks.
- Reinforced by: suggested bibliography and short professor's videos.
- assignments: professor's code recycling.
- On-line:
  - Remote collaboration and correction
  - Doesn't require powerful nor at campus computers
  - A dated record of each student's work

Inverted classroom

- Theory: emphasis on student's autonomus reading

Summary

A course centred on code

- Theory: text + equations + executable cod in digital notebooks.
- Reinforced by: suggested bibliography and short professor's videos.
- assignments: professor's code recycling.
- On-line:
  - Remote collaboration and correction
  - Doesn't require powerful nor at campus computers
  - A dated record of each student's work

Inverted classroom

- Theory: emphasis on student's autonomus reading
- Consultations: mostly on-line asynchronical and of public access

Summary

A course centred on code

- Theory: text + equations + executable cod in digital notebooks.
- Reinforced by: suggested bibliography and short professor's videos.
- assignments: professor's code recycling.
- On-line:
  - Remote collaboration and correction
  - Doesn't require powerful nor at campus computers
  - A dated record of each student's work

Inverted classroom

- Theory: emphasis on student's autonomus reading
- Consultations: mostly on-line asynchronical and of public access
- TA personal assisantace when completing assignments in synchronical meetings

Current developments

Current developments

2023 Students feedback improved:

Current developments

2023 Students feedback improved:

- Theory notes and code at repository

Current developments

2023 Students feedback improved:

- Theory notes and code at repository
- Grading of assignments methodology
  Evaluating each one of them $\rightarrow$ higher student's performance

Current developments

2023 Students feedback improved:

- Theory notes and code at repository
- Grading of assignments methodology
  Evaluating each one of them $\rightarrow$ higher student's performance

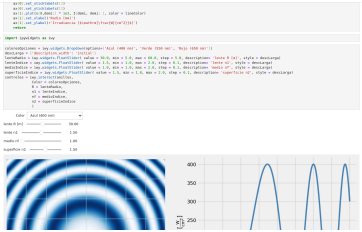
Current developments

2023 Students feedback improved:

- Theory notes and code at repository
- Grading of assignments methodology
  Evaluating each one of them → higher student's performance

2024

- A course on optics and waves will incorporate part of the methodology



Current developments

2023 Students feedback improved:

- Theory notes and code at repository
- Grading of assignments methodology
  Evaluating each one of them $\rightarrow$ higher student's performance

2024

- A course on optics and waves will incorporate part of the methodology
- AI assistance in code generation employing *GitHub Copilot*