A rational mechanics course where everything is made with Python code

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

Dept. of Engineering (DIIT), Universidad Nacional de La Matanza
Universidad Tecnológica Nacional
ARGENTINA

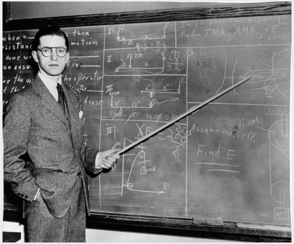
New Media Pedagogy 23

Value students and professors time

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

Value students and professors time

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

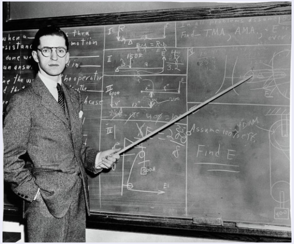


Most current classrooms: an excercise on transcription

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

Value students and professors time

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

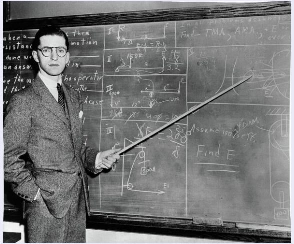


Most current classrooms: an excercise on transcription

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

Value students and professors time

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

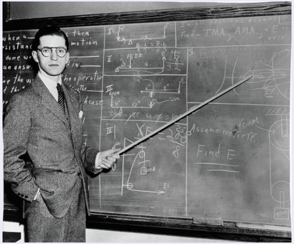


Most current classrooms: an excercise on transcription

- Professor: prepares lessons $\xrightarrow{\textit{by heart}}$ blackboard/slides
- Student: blackboard/slides $\xrightarrow{\textit{transcribes}}$ own notebooks
- Practice **reiterate** diagrams, calculations, etc.

Value students and professors time

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

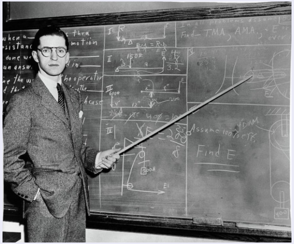


Most current classrooms: an excercise on transcription

- Professor: prepares lessons $\xrightarrow{\textit{by heart}}$ blackboard/slides
- Student: blackboard/slides $\xrightarrow{\textit{transcribes}}$ own notebooks
- Practice **reiterate** diagrams, calculations, etc.
- Boredom $\rightarrow$ loss of concentration on the subject

Value students and professors time

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

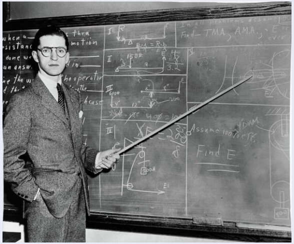


Most current classrooms: an excercise on transcription

- Professor: prepares lessons $\xrightarrow{\textit{by heart}}$ blackboard/slides
- Student: blackboard/slides $\xrightarrow{\textit{transcribes}}$ own notebooks
- Practice **reiterate** diagrams, calculations, etc.
- Boredom $\rightarrow$ loss of concentration on the subject

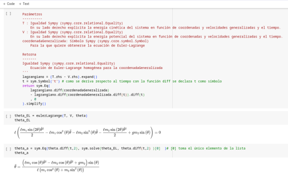
Value students and professors time

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



Most current classrooms: an excercise on transcription

- Professor: prepares lessons $\xrightarrow{\textit{by heart}}$ blackboard/slides
- Student: blackboard/slides $\xrightarrow{\textit{transcribes}}$ own notebooks
- Practice **reiterate** diagrams, calculations, etc.
- Boredom $\rightarrow$ loss of concentration on the subject

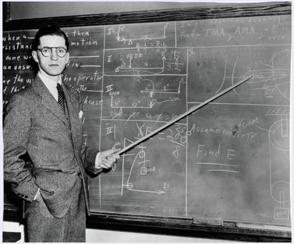


Digital course material with embedded code

- Professor: ideas $\xrightarrow{\textit{updates}}$ code/theory in repository

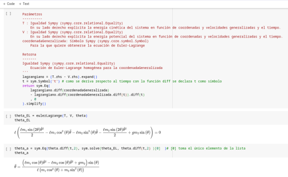
Value students and professors time

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



Most current classrooms: an excercise on transcription

- Professor: prepares lessons $\xrightarrow{\textit{by heart}}$ blackboard/slides
- Student: blackboard/slides $\xrightarrow{\textit{transcribes}}$ own notebooks
- Practice **reiterate** diagrams, calculations, etc.
- Boredom $\rightarrow$ loss of concentration on the subject



```
+ Code  + Text

[ ]     Parámetros
        ----------
        T : Igualdad Sympy (sympy.core.relational.Equality)
            En su lado derecho explicita la energía cinética del sistema en función de coordenadas y velocidades generalizadas y el tiempo.
        V : Igualdad Sympy (sympy.core.relational.Equality)
            En su lado derecho explicita la energía potencial del sistema en función de coordenadas y velocidades generalizadas y el tiempo.
        coordenadaGeneralizada: Símbolo Sympy (sympy.core.symbol.Symbol)
            Para la que quiere obtenerse la ecuación de Euler-Lagrange

        Retorna
        -------
        Igualdad Sympy (sympy.core.relational.Equality)
            Ecuación de Euler-Lagrange homogénea para la coordenadaGeneralizada
        '''
        lagrangiano = (T.rhs - V.rhs).expand()
        t = sym.Symbol('t') # como se deriva respecto al tiempo con la función diff se declara t como símbolo
        return sym.Eq(
            lagrangiano.diff(coordenadaGeneralizada)
            - lagrangiano.diff(coordenadaGeneralizada.diff(t)).diff(t)
            , 0
        ).simplify()

[ ] theta_EL = eulerLagrange(T, V, theta)
    theta_EL
```

$$\ell\left(\frac{\ell m_1 \sin(2\theta)\dot{\theta}^2}{2} - \ell m_1 \cos^2(\theta)\ddot{\theta} - \ell m_2 \sin^2(\theta)\ddot{\theta} - \frac{\ell m_2 \sin(2\theta)\dot{\theta}^2}{2} + g m_2 \sin(\theta)\right) = 0$$

```
[ ] theta_a = sym.Eq(theta.diff(t,2), sym.solve(theta_EL, theta.diff(t,2) )[0]  )# [0] toma el único elemento de la lista
    theta_a
```

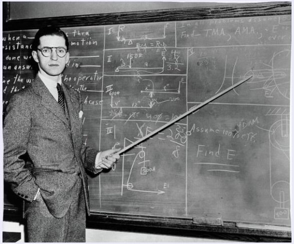
$$\ddot{\theta} = \frac{\left(\ell m_1 \cos(\theta)\dot{\theta}^2 - \ell m_2 \cos(\theta)\dot{\theta}^2 + g m_2\right)\sin(\theta)}{\ell\left(m_1 \cos^2(\theta) + m_2 \sin^2(\theta)\right)}$$

Digital course material with embedded code

- Professor: ideas $\xrightarrow{\textit{updates}}$ code/theory in repository
- Student: course repository $\xrightarrow{\textit{clones}}$ its own modifiable one

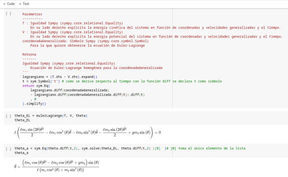
Value students and professors time

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



Most current classrooms: an excercise on transcription

- Professor: prepares lessons $\xrightarrow{\textit{by heart}}$ blackboard/slides
- Student: blackboard/slides $\xrightarrow{\textit{transcribes}}$ own notebooks
- Practice **reiterate** diagrams, calculations, etc.
- Boredom $\rightarrow$ loss of concentration on the subject



Digital course material with embedded code

- Professor: ideas $\xrightarrow{\textit{updates}}$ code/theory in repository
- Student: course repository $\xrightarrow{\textit{clones}}$ its own modifiable one
- **Recycles** professor's code to solve new problems

Engineering students can take advantage of code at every single lecture

Engineering students can take advantage of code at every single lecture

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

Engineering students can take advantage of code at every single lecture

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

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

DIIT

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

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

Engineering students can take advantage of code at every single lecture

- Currently they use a pocket calculator **after they learnt** learning arithmetics at school
- Now 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 can take advantage of code at every single lecture

- Currently they use a pocket calculator **after they learnt** learning arithmetics at school
- Now 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 can take advantage of code at every single lecture

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

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

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

- An example problem is solved by the professor provided code

Engineering students can take advantage of code at every single lecture

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

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

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

- An example problem is solved by the professor provided code
- The student **recycles** it to solve other related problems

Engineering students can take advantage of code at every single lecture

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

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

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

- An example problem is solved by the professor provided code
- The student **recycles** it to solve other related problems
- Gradually becoming autonomous by reusing not the provided but his own code

Tools — Jupyter notebook: text + equations + executable code

3. Obtenga una expresión para la tensión que ejerce la barra

$$Q_d = \lambda_1 \frac{\partial f_1}{\partial d} = \lambda_1$$

Por tanto hay que resolver el sistema con las 3 ecuaciones de Euler-Lagrange y la única de ligadura para determinar λ_1. Esta última hay que resolverla para su caso homogéneo y expersar su derivada segunda para que esté en el mismo orden que las de Euler-Lagrange, a fin de cuentas estamos resolviendo sistemas diferenciales de 2.o orden.

[14]:
```
f_1
```

[14]: $f_1 = -l + d$

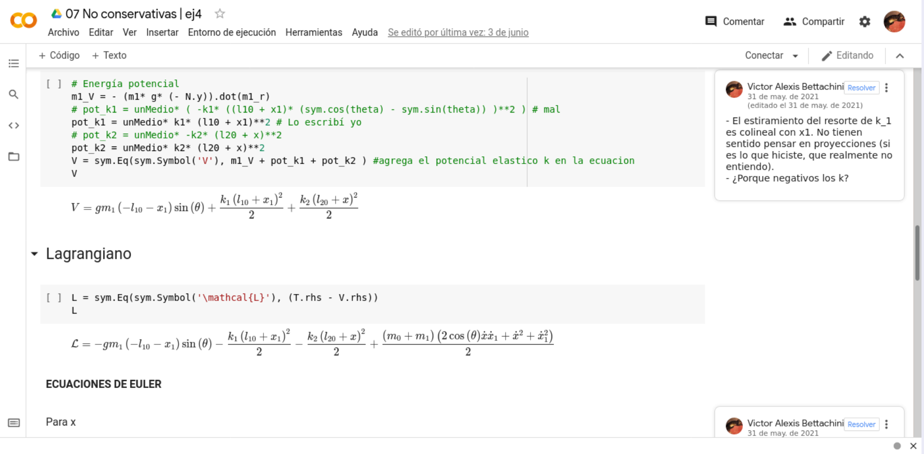
Determinamos también $\ddot{\theta}_1$ y $\ddot{\theta}_2$ pues serán necesarias para los cálculos numéricos posteriores.

[15]:
```
sistema = [theta1_EL.expand(),
           theta2_EL.expand(),
           d_EL.expand(),
           sym.Eq(f_1.rhs.diff(t,2), 0), # esto es igual a d punto punto = 0
          ]
variables = [theta1.diff(t,2), theta2.diff(t,2), lambda_1]
variables_sol = sym.nonlinsolve(sistema, variables).args[0]
```

[16]:
```
lambda_1_sol = sym.Eq(lambda_1, variables_sol.args[2])
lambda_1_sol.simplify()
```

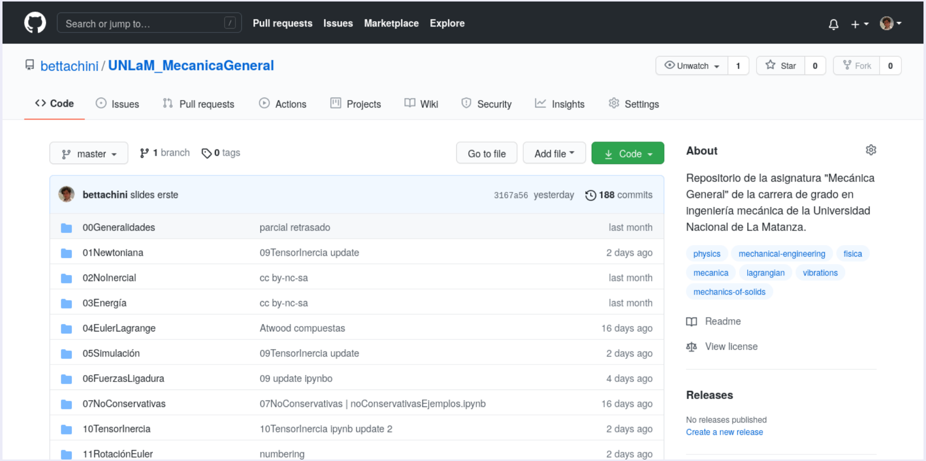
[16]:
$$\lambda_1 = \frac{m\left(2a\cos\left(\theta_1 - \theta_2\right)\dot{\theta}_1^2 + g\cos\left(2\theta_1 - \theta_2\right) + g\cos\left(\theta_2\right) + 2d\dot{\theta}_2^2 - 2\ddot{a}\right)}{\cos\left(2\theta_1 - 2\theta_2\right) - 3}$$

- The platform allows professors to edit and comment on the work of students.
- Students can collaborate remotely, working on the same Jupyter notebook.



Tools — All material in a GitHub repository

- Clearly organised, freely accesible, and easy to update.
- Google Colaboratory loads Jupyter notebooks directly from GitHub.



- LaTeX typesetting follows strictly the standards of the American Mathematical Society

Considero que el potencial V es nulo en el origen de coordenadas, es decir que donde se encuentra su mínimo $\varphi = 0, V(\varphi = 0) = -mg\ell$ y por tanto

$$V(\varphi) = mg\,(-\ell \cos\varphi) = -mg\ell \cos\varphi,$$

Como vemos la aproximación funciona bastante bien. Conformes con ella calculamos la fuerza

$$\vec{F} = -\vec{\nabla} V = -\left(\frac{\partial}{\partial r}, \frac{1}{r}\frac{\partial}{\partial \varphi}, \frac{\partial}{\partial z}\right) V(\varphi)$$

Pero solo nos interesa expresar la 2.a ley de Newton para lo que pasa en $\hat{\varphi}$

$$m\ddot{\vec{r}} \cdot \hat{\varphi} = -\frac{1}{r}\frac{\partial}{\partial \varphi} V(\varphi)$$

En el lado izquierdo de la expresión de la aceleración en cílindricas $\ddot{\vec{r}} = (\ddot{r} - r\dot{\varphi}^2)\hat{r} + (\dot{r}\dot{\varphi}^2 + r\ddot{\varphi})\hat{\varphi} + \ddot{z}\hat{z}$, nos quedamos solo con la componente en $\hat{\phi}$,

$$\ddot{\vec{r}} \cdot \hat{\varphi} = \dot{r}\dot{\varphi}^2 + r\ddot{\varphi}$$

y como el hilo del péndulo es rígido e inextensible $r \equiv \ell$ solo queda de esto

$$\ddot{\vec{r}} \cdot \hat{\varphi} = \ell\ddot{\varphi}$$

En el lado derecho la derivada del potencial respecto a φ es

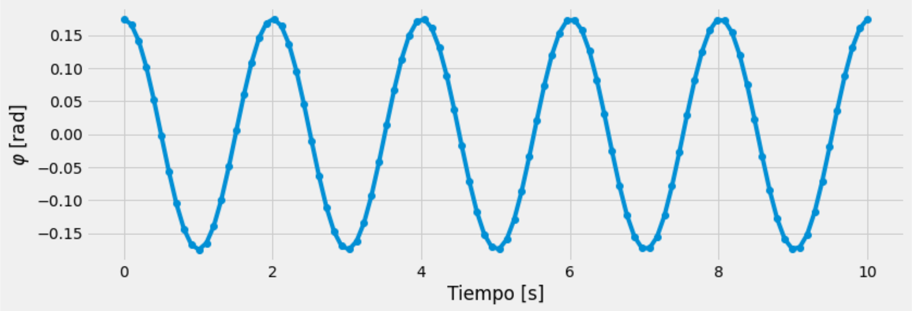
$$\frac{\partial}{\partial \varphi} V(\varphi) = mg\ell \sin(\varphi)$$

Tools — 1st class — Matplotlib: precise and reproducible graphics

- Explicit Python code for producing graphics. Students are enticed to play with it.

```
# graficación
fig, ax = plt.subplots(figsize=(12, 4))
ax.plot(tiempos, phi(tiempos), 'o-' )
ax.set_xlabel('Tiempo [s]')
ax.set_ylabel(r'$\varphi$ [rad]')
```

```
Text(0, 0.5, '$\\varphi$ [rad]')
```



Tools — 3rd class — SymPy: symbolic calculations

- Students effort centred on new subjects by freeing them of calculus and algebraic tasks.

[8]:
```
m2_v_cuadrado = m2_v.dot(m2_v)
m2_v_cuadrado
```

[8]: $\ell^2 \sin^2(\varphi)\dot{\varphi}^2 + (\ell\cos(\varphi)\dot{\varphi} + \dot{x})^2$

Con esto la energía cinética queda

$$
\begin{aligned}
T(\dot{x}_1, \varphi, \dot{\varphi}) &= \frac{m_1}{2}\left(\dot{\vec{r}}_1\right)^2 + \frac{m_2}{2}\left(\dot{\vec{r}}_2\right)^2 \\
&= \frac{m_1}{2}\dot{x}^2 + \frac{m_2}{2}\left(\dot{x}^2 + 2\dot{x}\ell\cos\varphi\dot{\varphi} + l^2\dot{\varphi}^2\right)
\end{aligned}
$$

[9]:
```
# Energía cinética
unMedio = sym.Rational(1,2) # Rational: fracción de enteros, alternativamente podría haberse usado 0.5
m1_T = unMedio* m1* m1_v_cuadrado
m2_T = unMedio* m2* m2_v_cuadrado
T = sym.Eq(sym.Symbol('T'), (m1_T + m2_T) ) # simplify: simplifica usando factor común y otras operaciones
T
```

[9]: $T = \frac{m_1\dot{x}^2}{2} + \frac{m_2\left(\ell^2\sin^2(\varphi)\dot{\varphi}^2 + (\ell\cos(\varphi)\dot{\varphi} + \dot{x})^2\right)}{2}$

Ecuaciones de Euler-Lagrange

Para x

[8]:
```
x_EL = sym.Eq(L.rhs.diff(x) - L.rhs.diff(x.diff(t)).diff(t), 0 ).simplify() # ecuación igualando a cero
x_EL
```

[8]:

$$m_1\ddot{x} + m_2\left(-\ell\sin(\phi)\dot{\phi}^2 + \ell\cos(\phi)\ddot{\phi} + \ddot{x}\right) = 0$$

Esta es una ecuación diferencial lineal de segundo orden homogena. De aquí podría despejarse $\ddot{x}$

[9]:
```
sym.Eq(x.diff(t,2),
       list( sym.solveset(x_EL, x.diff(t,2) ) )[0]  # solveset devuelve un set, que convertimos a lista
      ) # aceleración = x punto punto [m s-2]
```

[9]:

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

Pero queda en función de otra aceleración $\ddot{\phi}$.

Para ϕ

[10]:
```
phi_EL = sym.Eq(L.rhs.diff(phi) - L.rhs.diff(phi.diff(t)).diff(t), 0 ).simplify() # ecuación igualando a cero
phi_EL
```

- Mathematical complexity doesn't limit the scope of tackled mechanical problems.

[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]:

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

Tools — 5th class — SciPy: numerical computation of results

```
[22]: # defino una función con el sistema de derivadas
      # t : no se usa en este sistema pero lo dejamos para uso posterior
      # y : lista de estado con [y[0], y[1], y[2], y[3]]
      # y[0]: x
      # y[1]: x punto
      # y[2]: phi
      # y[3]: phi punto
      # dydt : lista de derivadas
      def y_punto(t, y):
          dydt = [y[1],
                  x_pp_numpy(y[0], y[1], y[2], y[3]),
                  y[3],
                  phi_pp_numpy(y[0], y[1], y[2], y[3]),
                 ]
          return dydt
```

```
[23]: # Integración de a pasos en el tiempo
      y_ode2 = solve_ivp(y_punto, (t_rango[0], t_rango[-1] ), y_inicial, t_eval = t_rango)
```

```
[25]: y_ode2.y[0]
```

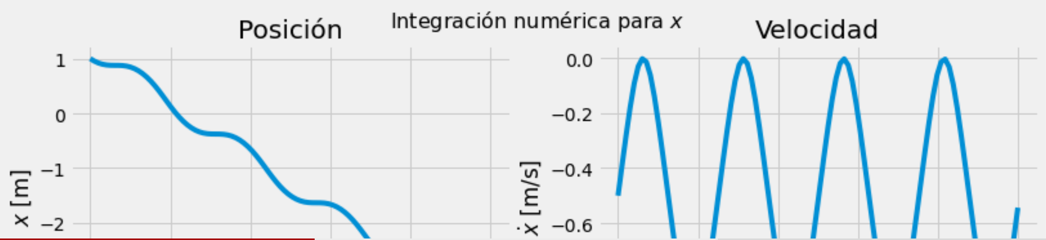
```
[25]: array([ 1.        ,  0.95510744,  0.92131146,  0.89820932,  0.88468059,
              0.87877042,  0.87745354,  0.87702754,  0.87352768,  0.86357726,
              0.84474673,  0.81565733,  0.77559949,  0.72423163,  0.66166451,
```

Tools — 5th class — Graphical analysis of numerical results

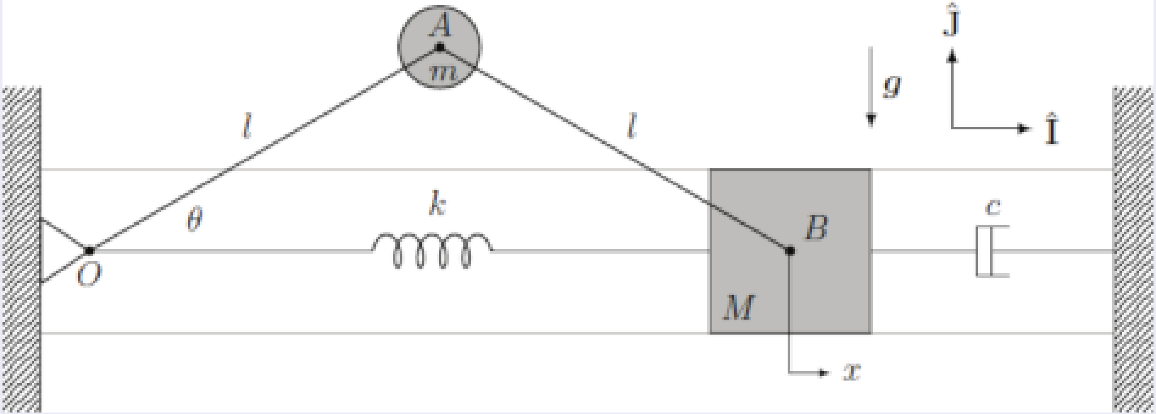
[26]:
```python
solucion = y_ode2
nombreCoordenada = 'x'

fig, ax = plt.subplots(nrows= 1, ncols= 2, squeeze=False, figsize=(12, 4)) # dos figuras en la misma fila
fig.suptitle('Integración numérica para $'+ nombreCoordenada + '$', fontsize=16)
ax[0,0].plot(solucion.t, solucion.y[0]) # posición x
ax[0,0].set(xlabel='t [s]', ylabel= '$' + nombreCoordenada+ '$ [m]', title='Posición')
ax[0,1].plot(solucion.t, solucion.y[1]) # velocidad x
ax[0,1].set(xlabel='t [s]', ylabel='$\dot{' + nombreCoordenada+ '}$ [m/s]', title='Velocidad')
```

[26]:
```
[Text(0.5, 0, 't [s]'),
 Text(0, 0.5, '$\\dot{x}$ [m/s]'),
 Text(0.5, 1.0, 'Velocidad')]
```

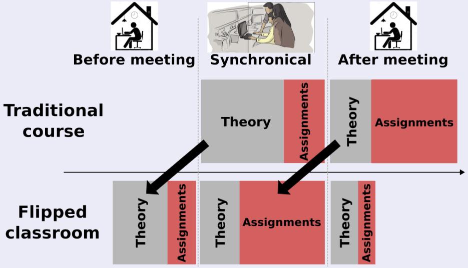


- Code from previous classes is **recycled** to model more realistic devices.



Methodology — Weekly cycle of our flipped classroom

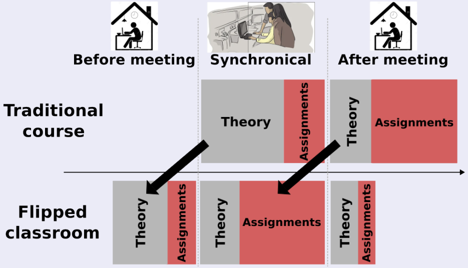
- New theory (Jupyter notebooks and videos) and assignments published on-line



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

Methodology — Weekly cycle of our flipped classroom

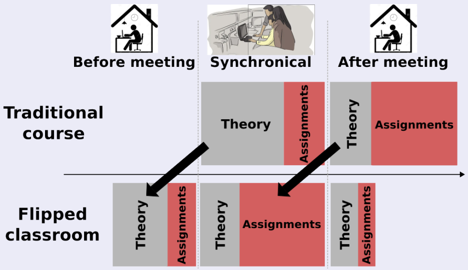
- New theory (Jupyter notebooks and videos) and assignments published on-line
- Previous week assignments **must** be turned-in for scoring



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

Methodology — Weekly cycle of our flipped classroom

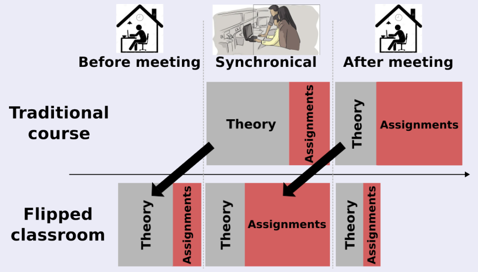
- New theory (Jupyter notebooks and videos) and assignments published on-line
- Previous week assignments **must** be turned-in for scoring
- On-line 24/7 **asynchronical** consultations that are **public** to other students



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

Methodology — Weekly cycle of our flipped classroom

- New theory (Jupyter notebooks and videos) and assignments published on-line
- Previous week assignments **must** be turned-in for scoring
- On-line 24/7 **asynchronical** consultations that are **public** to other students
- **synchronical** meetings with TA's to finish assignments



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

Tools — Google Colaboratory: asynchronic remote assistance

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
```

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

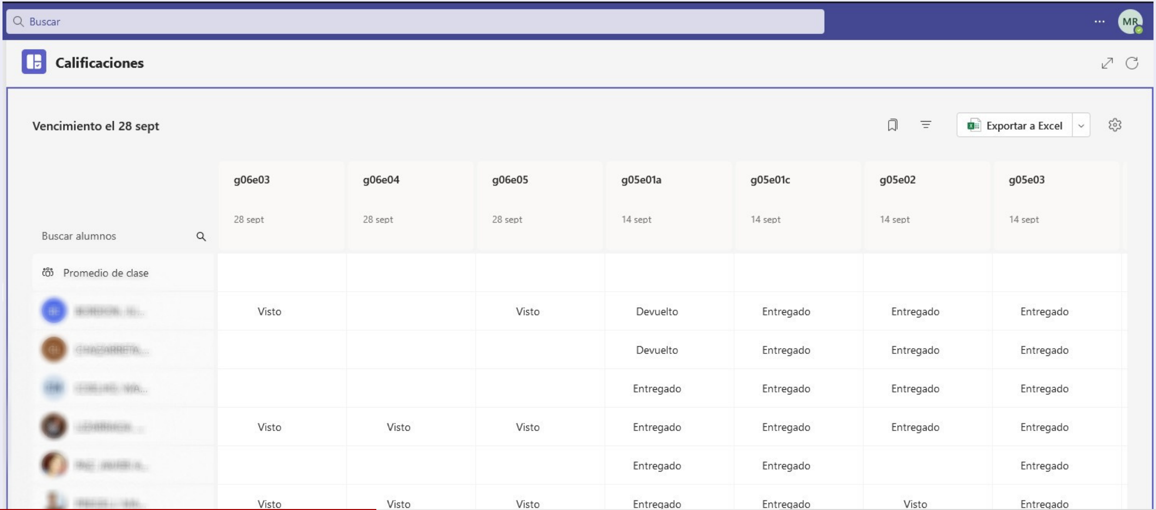
ECUACIONES DE EULER

Para x

Victor Alexis Bettachini Resolver
31 de may. de 2021

Tools — Microsoft Teams: individualized student follow-up

A record of the weekly turn-up of assignments



	g06e03 28 sept	g06e04 28 sept	g06e05 28 sept	g05e01a 14 sept	g05e01c 14 sept	g05e02 14 sept	g05e03 14 sept
Promedio de clase							
	Visto		Visto	Devuelto	Entregado	Entregado	Entregado
				Devuelto	Entregado	Entregado	Entregado
				Entregado	Entregado	Entregado	Entregado
	Visto	Visto	Visto	Entregado	Entregado	Entregado	Entregado
				Entregado	Entregado		Entregado
	Visto	Visto	Visto	Entregado	Entregado	Visto	Entregado

Course — Summary of methodology

A course centred on code

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

Flipped classroom

Course — Summary of methodology

A course centred on code

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

Flipped classroom

Course — Summary of methodology

A course centred on code

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

Flipped classroom

Course — Summary of methodology

A course centred on code

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

Flipped classroom

Course — Summary of methodology

A course centred on code

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

Flipped classroom

Course — Summary of methodology

A course centred on code

- Theory: text + equations + executable code 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

Flipped classroom

Course — Summary of methodology

A course centred on code

- Theory: text + equations + executable code 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

Flipped classroom

Course — Summary of methodology

A course centred on code

- Theory: text + equations + executable code 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

Flipped classroom

- Theory: emphasis on student's autonomus reading

Course — Summary of methodology

A course centred on code

- Theory: text + equations + executable code 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

Flipped classroom

- Theory: emphasis on student's autonomus reading
- Consultations: mostly on-line asynchronical and publicly accessible

Course — Summary of methodology

A course centred on code

- Theory: text + equations + executable code 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

Flipped classroom

- Theory: emphasis on student's autonomus reading
- Consultations: mostly on-line asynchronical and publicly accessible
- Synchronical mettings: TA's personal assistance for completing assignments

Course — Latest developments

2023: student's feedback improved:

- Repository material: theory notes and code revised from class to class
- Evaluation: grading each assignment led to a higher student's performance

2024:

- AI: students to code in *GitHub Codespaces* with *GitHub Copilot*
- Translation: repository content to English (collaborations welcome!)

GitHub Copilot AI assists with coding