

# Física 12.<sup>o</sup> ano

## 02 Análise de objectos em queda

*Versão: 0,8*

José Pedro do Amaral

10 de novembro de 2025

### Objectivos

O objectivo deste laboratório é o cálculo com exactidão da constante gravítica  $g$ . Para este efeito, usaremos primeiro o programa **Tracker** para determinar as posições de um objecto em queda livre. Depois, usaremos o programa **R** para ajustar um modelo às posições do objecto em queda e assim calcular  $g$  e o erro relativo ( $\varepsilon$ ) associado ao cálculo. Este programa será usado também para fazer um gráfico de qualidade do movimento de queda do objecto.

### Pré-requisitos

Não se aplicam se estiveres a usar um computador do colégio. Estes programas só deverão ser instalados se o computador ainda não os tiver. Se o computador já tiver o **Tracker** e o **R**, deverás começar no passo 1.

**Instalar o Tracker:** descarrega a versão para Windows a partir de <https://physlets.org/tracker/> e instala.

**Instalar o R:** descarrega a versão para Windows a partir de <https://cran.radicaldevelop.com/bin/windows/base/R-4.5.2-win.exe> e instala.

**Vídeo:** usa um vídeo nítido, com a câmara fixa e a bola bem visível. Inclui uma **régua** com comprimento conhecido no plano do movimento (idealmente, a régua deverá estar no mesmo plano que o movimento e assim uma régua vertical com um extremos marcados com uma cor óbvia seria ideal). Garante que sabes a taxa de fotogramas do vídeo (30 ou 240 fps são comuns).

**Ficheiro de vídeo compatível:** formatos comuns como **.mp4**, **.mov** ou **.avi** costumam funcionar bem. Se houver escolha, escolhe **.mp4**.

# Obtenção de dados

## 1. Abrir o vídeo no Tracker (MS Windows)

É de salientar que o colégio (e possivelmente a tua família) tem os computadores configurados em línguas diferentes: uns em português e outros em inglês. Estas instruções terão que ser ajustadas se o computador estiver em inglês (ou noutra língua). Isto geralmente é fácil de ser feito pois os vários comandos permanecem na mesma posição independentemente da língua escolhida.

1. **Para este laboratório:** o vídeo será **um** (1) dos que estão em <http://95.92.133.86/~amaral/>.
2. **Para descarregar o vídeo:** clica no vídeo e escolhe **Guardar vídeo como...** Grava o vídeo no **Ambiente de trabalho** (escolhe essa opção na janela abaixo de OneDrive).
3. Inicia o Tracker. Se o Tracker não estiver óbvio, deverás carregar na lupa e procurá-lo.
4. No Tracker, vai a **Ficheiro** → **Abrir** → **Seletor de ficheiros...** e escolhe o teu vídeo.
5. Na primeira abertura, confirma as **Ajustes do Corte de Vídeo...** (menu **Vídeo**):  
Verifica a **taxa de fotogramas (frame rate)** e a **escala temporal**. O Tracker lê normalmente isto do ficheiro; se souberes a taxa real (ex., 30 ou 240 fps são comuns no telemóvel), garante que os valores fazem sentido.  
Define o **intervalo temporal** a analisar (opcional e desnecessário se for um dos vídeos que forneci): arrasta as barras de início e de fim na linha temporal.

## 2. Definir a escala (calibração) e o referencial

### A) Escala (muito importante!)

1. No painel de ferramentas, cria um **Bastão de calibração** (ícone das cruzes azuis ligadas por uma linha com 1.0 ao lado ou **Trajetórias** → **Ferramentas de Calibração** → **Bastão de calibração**).
2. Arrasta as extremidades da régua virtual para coincidirem com duas marcas conhecidas no objeto real (por ex.: 1,00 m de uma régua física visível).
3. Introdus o **comprimento real** (ex.: 1,00 m ou o que eu disser que é o valor real para o vídeo escolhido) no campo do **Comprimento**.
4. Ajusta cuidadosamente as cruzes nas extremidades do metro que está na imagem. Quaisquer falhas de posição causarão um efeito nos cálculos.

## B) Referencial: eixos e origem

1. Cria um **eixo de coordenadas** (ícone magenta com ordenadas e abscissas ou **Triângulo azul invertido à direita por baixo da pasta amarela** → **eixos**).
2. Carrega em (0,0) e posiciona a **origem** num ponto útil (p. ex., no ponto de largada da bola ou no chão por baixo da trajetória). A escolha afectará o sinal dos valores obtidos. Eu recomendaria pôr o eixo das abscissas no pavimento onde a bola cai.
3. Verifica as **unidades**: depois da calibração, os eixos devem mostrar m em vez de px.

## 3. Criar a *trajetória* e marcar a posição da bola

1. Cria uma nova **Trajetoória** do tipo Massa Pontual: **Trajetoória** → **Novo** → **Massa Pontual**.
2. Prime as teclas **ctrl** e **shift**, o cursor deverá mudar para uma cruz de mira dentro de um círculo e enquanto as teclas estão carregadas:
  - (a) Clica no **centro** da bola, isto deverá activar a janela do **Autotracker**.
  - (b) Carrega no botão **Procurar**.
  - (c) Deixa correr a análise até ao fim.
  - (d) Ocasionalmente, o programa poderá perder a localização seguinte. Diz-me se isso acontecer.

## 4. Ver os dados ( $t, x, y$ ) e exportar para ficheiro

Para que a secção seguinte (Análise de Dados, secção 5) funcione de forma expedita, será necessário seguir estas instruções à risca. Garante que os nomes do ficheiro e que a sua pasta são exactamente os que peço.

1. A **tabela de dados** está por baixo do ícone com uma tabela que está do lado direito e perto da palavra *Dados*. Aqui poderás ver os resultados numéricos do seguimento. Os valores de  $t$  não são problemáticos, mas talvez tenham que ser corrigidos aquando da análise com o R na secção 5. Os valores de  $y$  deverão ser consistentes com uma largada a um pouco mais de 1,5 m. Os valores de  $x$  não serão usados.
2. **Exportar para ficheiro:**

**Ficheiro** → **Exportar** → **Ficheiro de Dados...**

Escolhe **Delimitador** como Separador.

Carrega em **Guardar como...**, dá o nome **bola.txt** ao ficheiro, garante que gravará em *Desktop* (poderá também ser chamado *Ambiente de Trabalho* ou *Secretária*) e carrega em **Save**.

3. Abre o ficheiro que acabaste de gravar e:
  - (a) confirma que as **unidades** exportadas são as esperadas (segundos e metros),
  - (b) confirma que o separador de decimais é a vírgula (,),

- (c) apaga a linha do topo que diz «Massa <uma letra>»,
  - (d) apaga a `tab` (invisível) que está à direita do `y`.
4. Grava as mudanças no ficheiro (`ctrl + g` ou equivalente).

## Análise de dados

### 5. R: carregar e analisar os dados

Aqui é que a porca torce o rabo! O R exige que os comandos passados estejam com a sintaxe correcta. Com calma, deverás ler os comentários (estão marcados com `#`) e executar os comandos em sequência. Quando escrevo *<qualquer coisa>* é porque não consigo prever que nome será. Logo que se saiba o nome, tudo deverá ser substituído pelo nome. Por exemplo, `"C:/Users/<utente>/Desktop"` será substituído por `"C:/Users/Aluno/Desktop"` se naquele portátil o utente for chamado Aluno. O símbolo  $\hookrightarrow$  indica somente que o que estiver a seguir é a continuação da linha anterior.

1. Carregar os dados e ficar com uma ideia do que se tem:

```
# Este comando desliga notação científica
options(scipen = 9999)

# Este comando mostra-te a directoria (pasta) corrente
# Lê aqui o nome do utente e confirma que pasta o R está lendo
# Por exemplo, fora o resultado "C:/Users/Aluno/Documents" quereria
   $\hookrightarrow$  dizer que o utente é Aluno e o R está a ler a pasta Documents
getwd()

# Este comando põe o R a trabalhar na secretária (Desktop)
# Substitui <utente> por Aluno (ou pelo nome que o comando anterior
   $\hookrightarrow$  tiver mostrado)
setwd("C:/Users/<utente>/Desktop")

# Este comando confirma que mudaste para a secretária e deverá
   $\hookrightarrow$  listar o ficheiro bola.txt que tem os dados
dir()

# Este comando carrega o ficheiro para um objecto de dados
bola_caindo<-read.table("bola.txt", header=TRUE, sep="\t", dec=",")

# Para saber a estrutura do objecto de dados
str(bola_caindo)

# Para ter uma ideia dos dados
plot(bola_caindo$t,bola_caindo$y)
```

2. Para analisar a primeira queda

```
# Ajustar o tempo para câmara-lenta
```

```

ajuste_vid <- 240/30

# Determinar por tentativa e erro onde começam e onde acabam os
↪ dados de interesse.
# O valor de 100 deverá ou não ser substituído por outro que permita
↪ mostrar a queda até ao primeiro contacto com o pavimento.
# Ajusta igualmente os valores de y1 e de t1.
y1<-bola_caindo$y[1:100]
t1<-bola_caindo$t[1:100]/ajuste_vid

# Ajustar o modelo
fit1 <- lm(y1 ~ I(t1^2))

# Extrair o parâmetro
m1 <- coef(fit1)[["I(t1^2)"]]

# Calcular g
(g1 <- 2*abs(m1))

# g de referência (Lisboa ao nível do mar)
g_ref <- c(9.8006)

# Erro relativo
(erro_rel1 <- abs((g_ref-g1)/abs(g_ref)))

```

3. E para acabar, um gráfico bonito:

```

# Gráfico
plot(t1,y1, ylim=c(0,1.5), type="n", ylab=c("y (m)"), xlab=c("t
↪ (s)"), main="bola em queda")

# Grelha discreta de coordenadas no fundo do gráfico.
grid()

# Linha pontilhada do modelo de queda.
lines(t1,predict(fit1, newdata=data.frame(seq(0,5,
↪ length.out=length(t1))))), lty=3)

# Pontos cor-de-rosa (subamostrados).
points(t1[seq(1, length(t1), length.out=15)],y1[seq(1, length(y1),
↪ length.out=15)], pch=21, bg="pink")

# Caixa de linhas mais grossas à volta do gráfico.
box(lwd=3)

# Etiquetas com os vários valores associados.
text(x = 0.1, y = 1.0, labels = bquote(bold(g)[calc] == .(round(g1,
↪ 3)) ~ scriptstyle("m/s"^2)), col = "black", cex = 1.2, adj = 0)

```

```

text(x = 0.1, y = 0.90, labels = bquote(bold(g)[ref] ==
↪  .(round(c(g_ref), 3)) ~ scriptstyle("m/s"^2)), col = "black",
↪  cex = 1.2, adj = 0)
text(x = 0.1, y = 0.80, labels = bquote(epsilon[rel] ==
↪  .(round(erro_rel1, 3))), col="black", cex=1.2, adj = 0)

```