

iemisc: Calculating the Reynolds number Examples

Irucka Embry, E.I.T. (EcoC²S)

2023-02-13

Contents

Replicate the R code	1
Problem 1 Statement	1
Solution 1	2
Problem 2 Statement	4
Solution 2	4
Works Cited	6
EcoC²S Links	6
Copyright and License	7

Replicate the R code

Note: If you wish to replicate the R code below, then you will need to copy and paste the following commands in R first (to make sure you have all the packages and their dependencies):

```
install.packages(c("install.load", "iemisc", "units", "round"))
# install the packages and their dependencies

# load the required packages
install.load::load_package("iemisc", "units", "round")
# load needed packages using the load_package function from the install.load
# package (it is assumed that you have already installed these packages)
```

Problem 1 Statement

Problem 17.2 [Lindeburg Practice]

“Points A and B are separated by 3000 ft of new 6 in schedule-40 steel pipe. 750 gal/min of 60°F water flows from point A to point B. Point B is 60 ft above point A.”

What is the Reynolds number?

From Appendix 16.B Dimensions of Welded and Seamless Steel Pipe [Lindeburg Manual], the internal diameter for a 6 inch nominal diameter new schedule-40 steel pipe is 0.5054 ft with an internal area of 0.2006 ft².

From Table 17.2 Values of Specific Roughness for Common Pipe Materials [Lindeburg Manual], the specific roughness, ϵ , for a steel pipe is 0.0002 ($2e - 04$) ft.

Solution 1

```
# 60 degrees Fahrenheit water new 6 in schedule-40 steel pipe determine the
# Reynolds number

# given the water flow of 750 gal / min create a numeric vector with the units
# of gallons per minute for the volumetric flow rate
Vdot <- set_units(750, gallon/min)
Vdot

## 750 [gallon/min]

# create a numeric vector with the units of cubic feet per second for the
# volumetric flow rate
Vdot <- Vdot

units(Vdot) <- make_units(ft^3/s)
Vdot

## 1.671007 [ft^3/s]

# given temperature of 60 degrees Fahrenheit create a numeric vector with the
# units of degrees Fahrenheit
T_F <- set_units(60, degree_F)

# create a numeric vector to convert from degrees Fahrenheit to Kelvin
T_K <- T_F
T_K

## 60 [degree_F]

# create a numeric vector with the units of Kelvin
units(T_K) <- make_units(K)
T_K

## 288.7056 [K]

# saturated liquid density at 60 degrees Fahrenheit (SI units)
rho_SI <- density_water(drop_units(T_K), units = "Absolute")

rho_SI <- set_units(rho_SI, kg/m^3)
rho_SI

## 998.9672 [kg/m^3]

# saturated liquid density at 60 degrees Fahrenheit (US Customary units)
rho_Eng <- density_water(drop_units(T_F), units = "Eng", Eng_units = "lbm/ft^3")

rho_Eng <- set_units(rho_Eng, lb/ft^3) # lbm/ft^3
rho_Eng
```

```

## 62.36349 [lb/ft^3]
# kinematic viscosity at 60 degrees Fahrenheit and density of rho (SI units)
nu_SI <- kin_visc_water(mu = dyn_visc_water(T = drop_units(T_K), units = "Absolute"),
  rho = density_water(T = drop_units(T_K), units = "Absolute"), rho_units = "kg/m^3",
  mu_units = "Pa*s or kg/m/s")

nu_SI <- set_units(nu_SI, m^2/s)
nu_SI

## 1.122882e-06 [m^2/s]
# kinematic viscosity at 60 degrees Fahrenheit and density of rho (US Customary
# units)
nu_Eng <- kin_visc_water(mu = dyn_visc_water(T = drop_units(T_F), units = "Eng",
  Eng_units = "lbf*s/ft^2"), rho = density_water(T = drop_units(T_F), units = "Eng",
  Eng_units = "lbm/ft^3"), rho_units = "lbm/ft^3", mu_units = "lbf*s/ft^2")

nu_Eng <- set_units(nu_Eng, ft^2/s)
nu_Eng

## 3.756632e-07 [ft^2/s]
# absolute or dynamic viscosity at 60 degrees Fahrenheit and density of rho (SI
# units)
mu_SI <- dyn_visc_water(T = drop_units(T_K), units = "Absolute")

mu_SI <- set_units(mu_SI, Pa * s)
mu_SI

## 0.001121723 [Pa*s]
# absolute or dynamic viscosity at 60 degrees Fahrenheit and density of rho (US
# Customary units)
mu_Eng <- dyn_visc_water(T = drop_units(T_F), units = "Eng", Eng_units = "lbf*s/ft^2")

mu_Eng <- set_units(mu_Eng, lbf * s/ft^2)
mu_Eng

## 2.342767e-05 [lbf*s/ft^2]
# create a numeric vector with the units of feet for the given specific
# roughness
epsilon <- set_units(2e-04, ft)
epsilon

## 2e-04 [ft]
# create a numeric vector with the units of feet for the given internal pipe
# diameter
Di <- set_units(0.5054, ft)
Di

## 0.5054 [ft]
# relative roughness (dimensionless) of the steel pipe
rel_roughness <- epsilon/Di
rel_roughness

```

```

## 0.0003957262 [1]
# internal area of the steel pipe
Ai <- Di^2 * pi/4
Ai

## 0.2006136 [ft^2]
# average velocity of the flowing water
V <- Vdot/Ai
V

## 8.329481 [ft/s]
# Reynolds number using the kinematic viscosity
Re_nu <- Re2(D = drop_units(Di), V = drop_units(V), nu = drop_units(nu_Eng))
Re_nu

## [1] 11206101
# Reynolds number using the absolute or dynamic viscosity
Re_mu <- Re1(D = drop_units(Di), V = drop_units(V), rho = drop_units(rho_Eng), mu = drop_units(mu_Eng),
  units = "Eng")
Re_mu

## $nu
## [1] 1.208658e-05
##
## $Re1
## [1] 348297

```

Michael Lindeburg calculated the Reynolds number as 3.4593114×10^5 [rounded to 3.46e05].

Problem 2 Statement

“This month’s problem asked what flow rate of water would be needed to have fully developed turbulent flow in a circular tube.” [Fosse]

What is the Reynolds number given the mass flow rate?

Solution 2

```

# given temperature of 22 degrees Celsius create a numeric vector with the
# units of degrees Celsius
T_C <- set_units(22, degree_C)
T_C

## 22 [°C]

```

```

# create a numeric vector to convert from degrees Celsius to Kelvin
T_K <- T_C
T_K

## 22 [°C]
# create a numeric vector with the units of Kelvin
units(T_K) <- make_units(K)
T_K

## 295.15 [K]
# saturated liquid density at 22 degrees Celsius (SI units)
rho_SI <- density_water(drop_units(T_K), units = "Absolute")

rho_SI <- set_units(rho_SI, kg/m^3)
rho_SI

## 997.7247 [kg/m^3]
# kinematic viscosity at 60 degrees Fahrenheit and density of rho (SI units)
nu_SI <- kin_visc_water(mu = dyn_visc_water(T = drop_units(T_K), units = "Absolute"),
  rho = density_water(T = drop_units(T_K), units = "Absolute"), rho_units = "kg/m^3",
  mu_units = "Pa*s or kg/m/s")

nu_SI <- set_units(nu_SI, m^2/s)
nu_SI

## 9.569716e-07 [m^2/s]
# absolute or dynamic viscosity at 60 degrees Fahrenheit and density of rho (SI
# units)
mu_SI <- dyn_visc_water(T = drop_units(T_K), units = "Absolute")

mu_SI <- set_units(mu_SI, Pa * s)
mu_SI

## 0.0009547942 [Pa*s]
# create a numeric vector with the units of inches for the given internal pipe
# diameter
Di <- set_units(4, inch)
Di

## 4 [inch]
# create a numeric vector with the units of meters for the given internal pipe
# diameter
Di <- Di
Di

## 4 [inch]
units(Di) <- make_units(m)
Di

## 0.1016 [m]
# given the mass flow rate of 0.765 kg/s (rounded in HTML) create a numeric
# vector with the units of kilograms per second for the mass flow rate

```

```
G <- set_units(0.76486004, kg/s)
G
## 0.76486 [kg/s]
Re3(D = drop_units(Di), G = drop_units(G), mu = drop_units(mu_SI), units = "SI")
## [1] 10038.96
```

Kendall Fosse provided 1e04 for the Reynolds number.

Works Cited

Kendall Fosse, “What rate is needed for turbulent flow? [Challenge Solved]”, ChEnected, <https://www.aiche.org/chenected/2010/10/what-rate-needed-turbulent-flow-challenge-solved>.

Michael R. Lindeburg, PE, *Civil Engineering Reference Manual for the PE Exam*, Twelfth Edition, Belmont, California: Professional Publications, Inc., 2011, page 17-4, 17-7, and A-22.

Michael R. Lindeburg, PE, *Practice Problems for the Civil Engineering PE Exam: A Companion to the “Civil Engineering Reference Manual”*, Twelfth Edition, Belmont, California: Professional Publications, Inc., 2011, pages 17-1 and 17-7.

The NIST Reference on Constants, Units, and Uncertainty, Fundamental Constants Data Center of the NIST Physical Measurement Laboratory, “standard acceleration of gravity g_n ”, <https://physics.nist.gov/cgi-bin/cuu/Value?gn>.

Wikimedia Foundation, Inc. Wikipedia, 15 May 2019, “Conversion of units”, https://en.wikipedia.org/wiki/Conversion_of_units.

EcoC²S Links

EcoC²S Home – <https://www.ecoccs.com/>

About EcoC²S – https://www.ecoccs.com/about_ecoc2s.html

Products – <https://www.questionuniverse.com/products.html>

EcoC²S Media – <https://www.ecoccs.com/media.html>

EcoC²S Resources – <https://www.ecoccs.com/resources.html>

R Trainings and Resources provided by EcoC²S (Irucka Embry, E.I.T.) – <https://www.ecoccs.com/rtraining.html>

Copyright and License

All R code written by Irucka Embry is distributed under the GPL-3 (or later) license, see the [GNU General Public License {GPL} page](#).

All written content originally created by Irucka Embry is copyrighted under the Creative Commons Attribution-ShareAlike 4.0 International License. All other written content retains the copyright of the original author(s).



This work is licensed under a [Creative Commons Attribution-ShareAlike 4.0 International License](#).