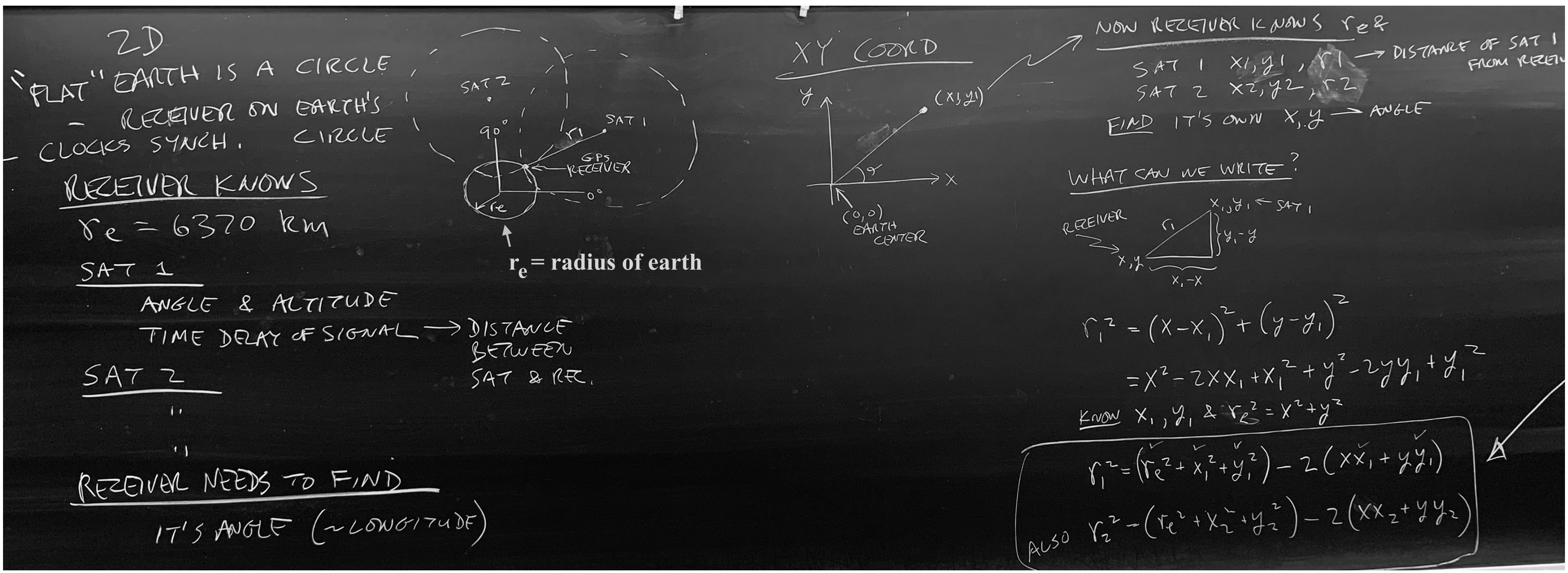
The Global Positioning System, GPS, is critical in our world today. It is a very interesting system and we can get a rough idea of how it works and also practice Matlab at the same time. First we looked at the GPS notes at

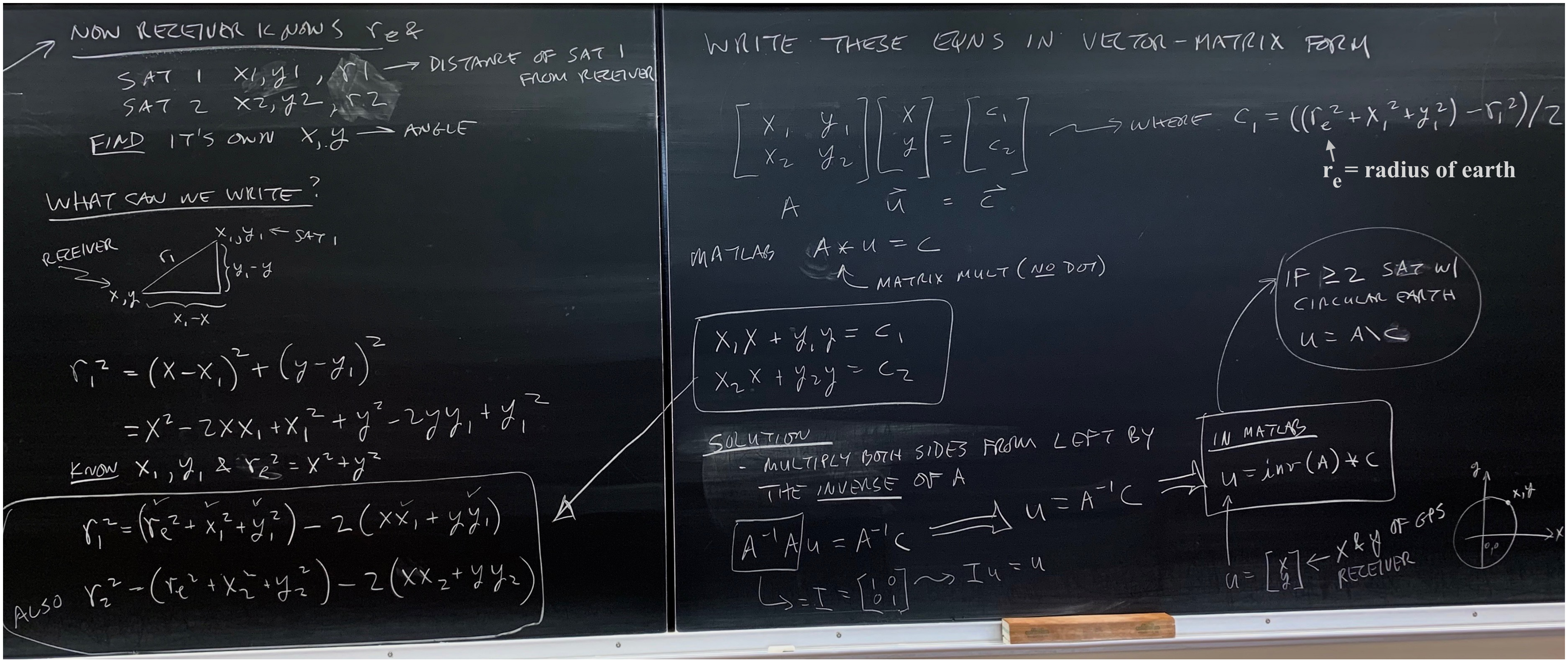
ReactorLab.net > Matlab > Notes & Examples > GPS, Global Positioning System, with Matlab

Then we worked on the solution of a simplified system: two satellites above a GPS receiver on the surface of a 2D, circular Earth, with all clocks synchronized.



This simple system ends up being a problem in linear algebra: solving simultaneous, coupled linear equations. When the number of equations (two, one for each satellite) equals the number of unknowns (two, the x and y location of the receiver on circular earth), then you can use the inverse of the coefficient matrix A. When you want to use more satellites to get a better estimate, then A is not square (# equations > # unknowns, an over determined system) and you then can use Matlab's symbolic matrix left division (\ ) operator to get the "least squares" best estimate of x and y.

For an example of matrix operations in chemical engineering see ReactorLab.net > Matlab > Notes & Examples > Array vs. Matrix operations



Now we get a little more realistic and consider GPS receiver on surface of a spherical Earth

Recommended book for those interested in GPS:
"Pinpoint: How GPS is Changing Technology, Culture, and Our Minds" by Greg Milner

SPHERICAL EARTH → RECEIVER ON SURFACE

NEED MINIMUM OF 3 SATELLITES

$$A = \begin{bmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ \vdots & & \\ x_m & y_m & z_m \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_m \end{bmatrix}$$

known satellite coordinates

$$c_1 = ((r_e^2 + x_1^2 + y_1^2 + z_1^2) - r_1^2)/2$$

r_e = radius of earth

r_1 = distance of sat 1 from receiver

3D x,y,z COORD OF GPS RECEIVER ON SURFACE OF EARTH

FOR $m = 3$

$u = \text{inv}(A) * c$

#rows MUST = #cols TO INVERT

FOR $m > 3$ → AN OVER DETERMINED LINEAR SYSTEM

$u = A \backslash c$ IN MATLAB

Earth is more similar to an oblate spheroid, https://en.wikipedia.org/wiki/Spheroid

GPS PROBLEM → SEE BLACKBOARDS OF AUG 29th

GIVEN

$$d = \begin{bmatrix} \# & \# & \# & \# \\ \# & \# & \# & \# \\ \vdots & \vdots & \vdots & \vdots \end{bmatrix};$$

$\leftarrow$ SAT 1

$\leftarrow$ SAT 2

latitude(°), longitude(°), altitude(km), & distance from receiver (km)

& GIVEN $r_e = \#$ (km)

r VECTOR

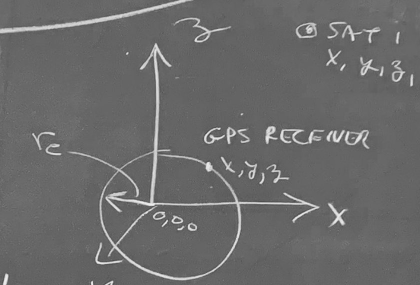
$A = \text{newLatLongToXYZ}(\_\_, r_e)$

$c_1 = ((r_e^2 + x_1^2 + y_1^2 + z_1^2) - r_1^2)/2$ Z

use Matlab “vectorization” to compute column vector c in one line of code - see option to sum each row in docs for sum at sum(A,dim)

$$\begin{array}{l} \text{SAT 1} \rightarrow \\ \text{SAT 2} \rightarrow \\ \\ \\ \end{array} \begin{bmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ \vdots & & \\ x_m & y_m & z_m \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_m \end{bmatrix}$$

UNKNOWN RECEIVER LOCATION



gps = newXYZtoLatLong(__, r_e)

↳ [lat, long, alt] SHOULD BE ~0

$A \quad * \quad u = c$

KNOWN UNKNOWN KNOWN

http://maps.google.com/maps?q=#,#

LAT LONG