final exam

Data types: Numeric data,Singles,Doubles, Integers,Character data,Logical data,Arrays of arrays (cell arrays and structure arrays), symbolic data

manipulate expression by hand: STEM courses evaluate u on how think,Never trust anything comes out of a computer.**Familiar with CAS:**Scientific expressions can become unwieldy to work by hand,Human mistakesCareless,Absent-minded,Easily distracted,Badhandwriting.

symbolic engine is *MuPad* by SciFace Software previously *MAPLE*; MuPad was discontinued as a stand-aloneprogram, and now is only available in MATLAB.

final exam (cont)

🏴: **declaring MATLAB symbols**:`y = sym('x')` give y a value of x ;`syms x` give x a value of x; `p = sym('a*(1-e^2)')` `Output: p = -a*(e^2 - 1)` ***expression:***can define a symbolcontaining symbols that are otherwiseunavailable in the Workspace; `my_p = sym('p == a*(1-e^2)')` `Output:p == -a*(e^2 - 1)` entire ***equation***

⬛ **`simplify`**:simplifyexpressions or equations using MuPad's rules(use `pretty` with `simplify`:easier to read); **`expand:`** multiplies out all of the parts of the expression or equation;**`factor`**: factors the expression or equation;**`collect`**: collects like terms;**`numden`**:find the numerator and denominator of an expression NOT equations; **`solve`**:(symbolicroot-finding) set the expression equal to zero and solve it, solve systems of linear or non❗ cannot solve higher-order systems with linear algebra, results are assignedin *alphabetical order*;**`subs`**: substituting# or other ; **`symfun`**: symbolic function; can usethe result to evaluate different inputs🏴:`syms x y`.

final exam (cont)

```
f = x^2 + x
g = symfun(f,y) ------- g(y) = x^2 + x;
```

`ezplot`:plotting for symbolic expression, need a defult range🏴:

```
f = sym('x^2')
ezplot(f, [-10,10])
```

⬛:**Derivative**: instantaneous time rate of change of a slope; an analogous word is differential**`diff(f)`**calculates the symbolic first derivative of a symbolic functionwith respect to the default independent variable;**`diff(f,symvar)`** calculates the symbolic first derivative of a symbolicfunction with respect to the default independent variable **`symvar`**❗ (symvar has tobe in single quotes if the variable does not already exist as a symbolic

final exam (cont)

variable); **`diff(f,n)`** calculates the symbolic nth derivative of the symbolic function with respect to the default independent variable;diff(f,symvar,n) or **`diff(f,n,symvar)`**calculates thesymbolic nth derivative of the symbolic function f with respect to the symvar----; **Integral**: the integral represents the area under a curve and above**`int(f)`** calculates the symbolic single integral of a symbolic function with respect to the default independent variable;**`int(f,symvar)`**;**`int(f,a,b)`** evaluates theresults of the integral over the symbolic or numeric range; **`int(f,symvar,a,b)`**;**Differential Equation (DE):** An equation containing an unknown function and itsderivatives; **`dsolve`**:calculate solutions to differential equations; **`D`**specify derivitive if you need to specify a nth order derivative, specify n afterthe symbol D ex: **`D4y`**; **`dsolve(equation)`**;**`dsolve(equation,symvar)`** ; **`dsolve(equation,condition1,condition2,…,conditionN, symvar)`** .

final exam (cont)

❗diff can also be used to calculate the differences then finding slope between points..

Converting Symbolic Expressions to Anonymous Functions

a. Only available starting in versions of MATLAB starting with version 2007B (this is one of the features that was incorporated with the adoption of MuPad)

b. To create anonymous symbolic functions, use the `matlabFunction`{{fa-flag}}

```
syms x
y = x^2 - x + 1
dy = diff(y)
f = matlabFunction(dy)
f(1)
```

⬛: **Interpolation**: consists of "method[s] of constructing new data points within the range of a discrete set of known data points. ***interp1***;***yi = interp1(x,Y,xi)*** Interpolates to find yi, the interpolated function values at the points in the vector or array xi.x contains your known data points (whose functions values are Y),which must be a vector, though xi can be a scalar, vector, ormultidimensional array. yi will always be the same size as xi;***yi = interp1(Y,xi)*** x = 1:N, where N is length(Y) (for a vector)or size(Y,1) (for a matrix);***yi = interp1(x,Y,xi,method)***; ***yi = interp1(x,Y,xi,method,'extrap')***.

final exam (cont)

Extrapolation consists of "the process of estimating,beyond the original observation range, the value of a variable on the basis of its relationship with another variable.❗Some interpolation are excellent,yielding useful results BUT extrapolation can be a fool's errand. **Linear Interpolation**:points (1, 3) and (-2, 5) Use linear interpolationto find estimate the y-value at the point x = −0.5?

(slope intercept form)$y = mx + b$----------m=5-3/-2-1=2/-3=y=-2/3x+b----------to find b: Substitute in one of points $y = mx + b$= 5=−2/3(−2) +b→b=3 2/3 ----------the line is y =-2/3 x + 3 2/3------ evaluate it at x = −0.5 to find y(−0.5): y −0.5 = −2/3 (−0.5) +3

y(−0.5)=4

Approch(pointslopeform) 2:$y - y1 = m\ (x - x1)$

find the slope m=-2/3

y−3=−2/3(x−1)→y−3=−2/3 x +2/3-------y=−2/3x+2/3+3 →y=−2/3x+3 2/3

the line is y =-2/3 x + 3 2/3 evaluate it at x = −0.5 to find y(−0.5): y −0.5 = −2/3 (−0.5) +3

y(−0.5)=4

❗:**Linear Interpolation**easy to do, BUT NOT best go-to solution if need accuracy.**Spline Interpolation**:A spline is to use a different polynomial between each pair of discrete points.**Cubic splines** correct for this flaw by ensuring that at the data points, the adjacent splines have the same 0th, 1st and 2nd derivatives; **Curve fitting** is "the process of constructing a curve, or mathematical function, that has the best fit to a series of data points, possibly subject to constraints ***polyval***, ***polyfit***; ❗ / operator use it to solve least squared problems(less error). ❗ Goal is the minimize the**residuals**: the difference between the actual and predicted values at a given point (i.e.

final exam (cont)

the error).take its derivative and look to see where it is zero (this gives us the **extrema** – the extreme points of the function) to find the minimum of function; ❗ numerical differentiation tends to amplify noise.**Taylor series** is a series expansion of a function $f(x)$ about a given point a.special case, known as **Maclaurin series**, of the Taylor series exists in which $a = 0$. ❗ The “Big O” notation (asymptotic notation) indicates higher order terms (H.O.T.s). **Central Difference:** `gradient`; **Root Finding**: zero a function; `fzero`.. ■: **finding area under curve**: Rectangles,Trapezoids,Parabolas. **Riemann Sums** :❗ If you average the left and right Riemannsum, you get the trapezoidal sum.**Left Riemann Sum**:fits rectangles underneath curve using left of interval as location for hight of rectangle; Overestimate if f decreasing & vice.versa.**Right Riemann Sum**: like left Riemann Sum but in right instead; Overestimate if f is increasing and vice versa.**Middle Riemann Sum**: Approximates the function by its value at themiddle point of the subinterval, yieldingmultiple rectangles with a base of Δx and the average height between the left and right.❗ This better than R & L Riemann sum.**Trapezoidal Rule**:Approximates the function by fitting trapezoids underneath the curve.**Simpson’s Rule**: Approximates the function by fitting parabolas under the curve.

final exam (cont)

❗ : must use an even number of intervals; Pros and cons of using this versus trapezoids – more computationally expensive, but a better fit at times.❗ Left-point and right-point sums were just wrong ❗mid-point, trapezoidal and Simpson’s) all got the correct answer. The choice between these depends on what the data looks like and what computational expense you can tolerate.**solve differentialequations**:Euler’s Method(forward Euler method), Runge-Kutta methods.

final exam (cont)

“state-space: breakyour system down into a system of simultaneous first order differential equations. ❗ The number of first order equations will be equal to the sum of number of independent variable(s) times the order..

tables

Symbolic Command	Description	Symbolic Command Description Numeric Analog
ezplot	2D plot	plot
ezmesh	Wireframe mesh	mesh
ezmeshc	Contour plot under wireframe mesh	meshc
ezsurf	Surface plot	surf
ezsurfc	Contour plot under surface plot	surfc

tables (cont)

ezcontour	Contour plot	contour
ezcontourf	Filled contour plot	contourf
ezplot3	3D plot	plot3
ezpolar	Polar plot	polar