

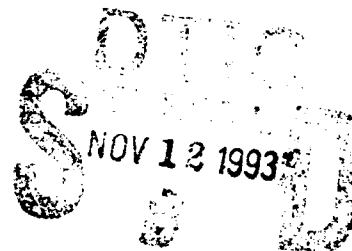
AD-A272 500



2

NAVAL POSTGRADUATE SCHOOL

Monterey, California



THESIS

USE OF COMPUTERS IN THE INSTRUCTION
OF INTEGRAL CALCULUS

by

Dennis Arthur Polaski

June 1993

Thesis Co-Advisors:

Beny Neta
David Canright

Approved for public release; distribution is unlimited

93-27646



REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE <p style="text-align: center;">March 1993</p>	3. REPORT TYPE AND DATES COVERED <p style="text-align: center;">Master's Thesis</p>
----------------------------------	---	--

4. TITLE AND SUBTITLE <p style="text-align: center;">USE OF COMPUTERS IN THE INSTRUCTION OF INTEGRAL CALCULUS</p>	5. FUNDING NUMBERS
---	--------------------

6. AUTHOR(S) <p style="text-align: center;">Polaski, Dennis Arthur</p>	
---	--

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) <p style="text-align: center;">Naval Postgraduate School Monterey, CA 93943-5000</p>	8. PERFORMING ORGANIZATION REPORT NUMBER
--	--

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSORING MONITORING AGENCY REPORT NUMBER
---	--

11. SUPPLEMENTARY NOTES

The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION / AVAILABILITY STATEMENT <p style="text-align: center;">Approved for public release, distribution is unlimited.</p>	12b. DISTRIBUTION CODE
--	------------------------

13. ABSTRACT (Maximum 200 words)

One of the most difficult concepts in the area of integral multivariate calculus is finding the limits of integration. This thesis describes an interactive computer program designed to help students understand this important concept. The program shows how a given domain is plotted and teaches how to find the limits of integration when evaluating two-dimensional integrals.

The program allows the user to enter any known information about a region and then evaluates the integral. The region is plotted, the limits of integration are given along with the area of the region. The program handles cartesian and polar coordinate, two-dimensional integral problems. This program could be used independently by the student and/or used by the calculus instructor in the classroom.

14. SUBJECT TERMS <p style="text-align: center;">Double Integrals, Integral Calculus</p>	15. NUMBER OF PAGES <p style="text-align: center;">165</p>
---	---

	16. PRICE CODE
--	----------------

17. SECURITY CLASSIFICATION OF REPORT <p style="text-align: center;">Unclassified</p>	18. SECURITY CLASSIFICATION OF THIS PAGE <p style="text-align: center;">Unclassified</p>	19. SECURITY CLASSIFICATION OF ABSTRACT <p style="text-align: center;">Unclassified</p>	20. LIMITATION OF ABSTRACT <p style="text-align: center;">UJ</p>
--	---	--	---

Approved for public release; distribution is unlimited.

**USE OF COMPUTERS IN THE INSTRUCTION OF
INTEGRAL CALCULUS**

by

Dennis Arthur Polaski
Captain, United States Army
B.S., United States Military Academy, 1983

Submitted in partial fulfillment
of the requirements for the degree of
MASTER OF SCIENCE IN APPLIED MATHEMATICS

from the

NAVAL POSTGRADUATE SCHOOL
June 1993

Author:



Dennis Arthur Polaski


Approved by:



Beny Neta, Co-Advisor



David Canright, Co-Advisor



Richard Franke, Chairman
Department of Mathematics

PREFACE

Upon completion of my graduate work at the Naval Postgraduate School I will be assigned as an Assistant Professor in the Department of Mathematical Sciences, at the United States Military Academy. While there I will teach the cadets calculus. I will incorporate this package on integral calculus that I have developed into the cadets' instruction.

As an instructor I will see this integral calculus package used by the cadets. I will get feedback from them on its use and its value in the instruction. I will then use the feedback to make improvements to the package and to expand its capabilities. It is my vision that such an instructional aid will improve the overall quality and effectiveness of the calculus instruction at the Military Academy.

DTIC QUALITY INSPECTED 4

Accession For	
Microfilm	<input checked="" type="checkbox"/>
microfiche	<input type="checkbox"/>
Unannounced	<input type="checkbox"/>
Continuation	
By	
Distribution	
Availability	
Date	
A-1	

ABSTRACT

One of the most difficult concepts in the area of integral multivariate calculus is finding the limits of integration. This thesis describes an interactive computer program designed to help students understand this important concept. The program shows how a given domain is plotted and teaches how to find the limits of integration when evaluating two-dimensional integrals.

The program allows the user to enter any known information about a region and then evaluates the integral. The region is plotted, the limits of integration are given along with the area of the region. The program handles cartesian and polar coordinate, two-dimensional integral problems. This program could be used independently by the student and/or used by the calculus instructor in the classroom.

THESIS DISCLAIMER

The reader is cautioned that computer programs developed in this research may not have been exercised for all cases of interest. While effort has been made, within the time available, to ensure that the programs are free of computational and logic errors, they cannot be considered validated. Any application of these programs without additional verification is at the risk of the user.

TABLE OF CONTENTS

I.	INTRODUCTION	1
II.	MOTIVATION	7
III.	USE OF PACKAGE IN CALCULUS INSTRUCTION	12
IV.	WHY MATHEMATICA	15
V.	STRUCTURE OF THE PROGRAM	17
VI.	SUBPROGRAM GRAPH	21
VII.	SUBPROGRAM CART	23
VIII.	SUBPROGRAM INVERSE	31
IX.	SUBPROGRAM CARTPLOT	36
X.	SUBPROGRAM POLAR1	39
XI.	SUBPROGRAM POLPLOT1	43
XII.	SUBPROGRAM POLAR2	46
XIII.	SUBPROGRAM POLPLOT2	50
XIV.	SUBPROGRAM EXAMPLES	53
XV.	FUNCTIONS AND NOTATION	54
XVI.	FUTURE WORK	56
	APPENDIX A LAB BOOK	58
	APPENDIX B TUTORIAL	94
	APPENDIX C PROGRAM CODE	105
	LIST OF REFERENCES	157
	INITIAL DISTRIBUTION LIST	158

I. INTRODUCTION

In the area of integral multivariate calculus one of the most difficult topics is deciding on the limits of integration. This package shows the calculus student how a given domain is plotted and teaches the student how to find the limits of integration when evaluating two-dimensional integrals.

Integral calculus is the mathematics we use, for example, to find lengths and areas of irregular shapes; to calculate average values of functions; and to predict future values in population sizes and future costs of living. The development of integral calculus starts from the calculation of areas. The limits used to define areas are special cases of a definite integral.

The focus of the package is on teaching the student how to correctly find the limits of integration for a region. While for typical calculus problems the limits of integration are determined analytically, it is quite helpful to see the regions of interest. It then is important to be able to accurately graph the regions. The calculus student may have difficulty with curve sketching especially in polar coordinates. This package allows the student to visualize the region of interest enabling the student to better understand

the problem. It is essential that the student be able to see accurate plots.

This package was developed using the **Mathematica** software. **Mathematica** uses an adaptive sampling algorithm for plotting to determine when and where to sample a function. A section with large curvature is sampled more frequently than a flat section of a function. This sampling algorithm coupled with a set prescribed number of equally spaced samples produces the quite reasonable plots we want. This graphing capability is one of the greatest attributes of this package.

This package, which teaches the student how to find the limits of integration, has two parts. The first part, which is found in Appendix A, consists of a lab book. This lab book addresses two-dimensional integrals and walks the student through numerous example problems. The two-dimensional case considers regions given in the cartesian coordinate system as well as regions in the polar coordinate system. In all examples the student is shown the plot of the functions and then is shown how the limits of integration are determined. The integrals are set up and evaluated and the area of the region is given.

The second part of this package is an interactive program written in **Mathematica's** programming language. With this program the student can solve two-dimensional integral problems, given in both the cartesian coordinate system and the polar coordinate system. The student enters the functions

which define a region's boundaries and any other known information about the region. The curves are then plotted. The limits of integration are determined and the area of the region is calculated. The program output includes the limits of integration, the curves and the calculated area of the regions. The program also can be used to display any of the plots for the example problems found in the lab book.

The program's greatest value, however, is in allowing the user to input his own problems. The program is designed to handle those types of integral calculus problems found in any introductory level calculus text book.

No knowledge of **Mathematica** is required to use the interactive program. The tutorial, found in Appendix B, tells the user how to load the program and how to use it. The tutorial takes the user through three example problems. The first is an integral problem given in cartesian coordinates. The second problem is in polar coordinates and finds the area between a polar curve and the origin. The final problem is also in polar coordinates and finds the area between two polar curves. These three problems are simple problems that show how the program is used. The tutorial, while not comprehensive, gives the user some idea of the types of problems the program handles. It demonstrates how the program prompts the user for input and how that input is entered. The program is capable of much more complicated problems than those in the tutorial. In general, once the package is loaded and the

program is started the user simply follows the instructions that the program displays.

Due to the variety of ways in which a region can be defined or in which a problem can be worded the program must be versatile. The program prompts the user in such a way as to solicit the necessary information about the region regardless of how the problem is stated. As mentioned previously the program is written to handle the types of problems found in calculus text books. These problems are generally well defined such that the program has no difficulty in determining the region of interest. On the other hand it is quite possible to contrive problems which are not well defined and for which the program is unable to determine the region. An example of this type is when there is not enough information provided. It is also possible to have a problem which is seemingly well defined for which the program does not work. An example of this type of problem is in cartesian coordinates when the functions intersect and do not define a single unique region. For instance the functions $y=12-x^2$, $y=x$ and $y=0$ intersect to form three separate regions (see Figure 1, page 5).

If such a problem is entered, the program produces either no output, bad output, or an error message. There is still a way in which the program can be used to solve these problems. For problems given in cartesian coordinates there is a cartesian plot-only option. If this option is selected the

user can input all the functions for specified ranges of the variables. The functions will then be displayed and the plot can be used to estimate the limits of integration. With the estimated limits of integration obtained from the plot of the functions the user can run the program again with this new information. This plot-only option can be used also if the user just wants to see what a particular function looks like.

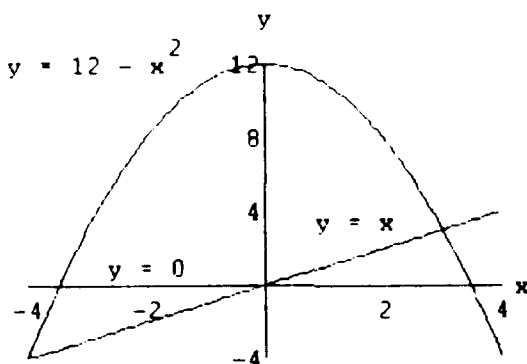


Figure 1

In general the program does not have any difficulty with the plots of the functions given in polar coordinates. If the limits of integration are not known in advance the program plots the polar functions from $Q=0$ to $Q=2\pi$ radians where Q represents the polar angle theta. For the two options, finding the area between a polar curve and the origin and finding the area between two polar curves, this feature acts as a plot-only option. The values of Q for which the polar functions intersect are displayed when finding the area between two curves. The values of Q for which the polar function is equal to zero are displayed when finding the area between a polar curve and the origin.

However if the user only wants to plot the polar functions without solving the integral problem, he can, both for plotting the area between a polar curve and the origin and for plotting the area between two polar curves.

II. MOTIVATION

Integral calculus has a wide variety of applications in mathematics, physics, engineering, economics, medicine, and other fields. Integrals are used, for example, to compute areas, volumes, lengths of curves, forces of fluids, work, weight, centers of gravity, and probabilities, and to determine functions from their rates of change. In these many applications often the most difficult part is in properly setting up the integrals to be evaluated. More exactly the difficulty lies in determining the limits of integration for these integrals.

Let R denote the region in the xy -plane that is bounded on the top by the graph $y=f(x)$ and on the bottom by the graph $y=g(x)$ and that extends from $x=a$ on the left to $x=b$ on the right. Double integrals over R may be expressed as iterated integrals by the formula

$$\iint_R h(x,y) dA = \int_{x=a}^{x=b} \left[\int_{y=g(x)}^{y=f(x)} h(x,y) dy \right] dx. \quad (1)$$

To evaluate the iterative integral we start at the innermost integral and work toward the outside.

The most difficulty with evaluating two-dimensional integrals given in cartesian coordinates is in determining $g(x)$, $f(x)$, a , and b which are limits of integration. The region R may have either changing upper or lower boundaries in

which case the region R is composed of subregions. The double integral over R then becomes equal to the summation of double integrals, that are over each of the subregions. Then for each subregion the limits of integration must be determined. This situation is more complicated, making determining the limits of integration difficult.

The focus of this thesis is teaching how to find the limits. As such, all the two-dimensional integral problems in this thesis are to find the area of a given region. That is $h(x,y)=1$ and so we have for integrals given in cartesian coordinates

$$\text{Area} = \int_{x=a}^{x=b} \left[\int_{y=g(x)}^{y=f(x)} dy \right] dx. \quad (2)$$

After evaluating the innermost integral the equation becomes

$$\text{Area} = \int_{x=a}^{x=b} [f(x) - g(x)] dx. \quad (3)$$

The functions $y=f(x)$ and $y=g(x)$ can be found by graphing the given functions. If the left limit, $x=a$, and the right limit, $x=b$, are not known they are determined analytically. The functions $y=f(x)$ and $y=g(x)$ are equated and solved for x .

Similarly in polar coordinates we have

$$\text{Area} = \iint_R r dA = \int_{Q=a}^{Q=b} \left[\int_{r=r_1(Q)}^{r=r_2(Q)} r dr \right] dQ \quad (4)$$

where both r_1 and r_2 are functions of Q . For finding the area between a polar curve and the origin, $r_1=0$ and we have

$$\text{Area} = \int_{Q=a}^{Q=b} \left[\int_0^{r=r_2(Q)} r dr \right] dQ. \quad (5)$$

If we evaluate the innermost integral we get

$$\text{Area} = \int_{Q=a}^{Q=b} \frac{1}{2} r_2^2 dQ. \quad (6)$$

For finding the area between two polar curves we evaluate the innermost integral in equation (4) above to get

$$\text{Area} = \int_{Q=a}^{Q=b} \frac{1}{2} [r_2^2 - r_1^2] dQ. \quad (7)$$

Again the most difficulty with two-dimensional integrals given in polar coordinates is setting up the integrals. More specifically the difficulty is in determining the elements a , b , r_1 , and r_2 which are the limits of integration. In polar coordinate integral problems the graphs of the polar curves are used to find the limits of integration. In finding the area between a polar curve and the origin the limits a and b (if not known) are determined by solving the equation $r_2=0$ for r . In finding the area between two polar curves the limits a and b (if not known) are determined by equating r_1 and r_2 and solving for r .

The calculus student must do numerous example problems in order to become proficient and confident in properly setting up an integral. The more problems a student sees and does, the more confident the student becomes. This package provides numerous worked solutions to problems. The program provides the solutions to as many integral problems as the student may wish to work. The program can be used with whatever calculus textbook the student may be using.

The program gives solutions that include not only the numerical area, but also the graph of the region and the limits of integration. The students then can use this information to identify their problem areas. The students may find that they are getting the right limits of integration when they work the problems but that their solutions are not correct. Then it is likely that they are making mistakes when evaluating the integral. In this case the students see that they should review their integration techniques. They may be finding incorrect limits of integration. Then the graph of the region may be helpful in determining where the errors are being made. At any rate the program provides a great deal of useful information. Most textbooks provide just the value of the integral.

The lab book contains numerous worked out example problems. It will complement the few worked problems that are likely to be found in most textbooks. The fact that the package is adaptable to any calculus textbook is very useful.

The simple-to-use program gives the students an extremely efficient and effective way of learning how to find limits of integration. The use of mathematical software can also make learning calculus more enjoyable. Typically students are first introduced to mathematical software in a higher level course. By using this package in their calculus instruction the students are introduced to mathematical software early during their education. In this age of computers this becomes more and more important all the time.

At the Naval Postgraduate School there is currently no introduction to mathematical software included in the syllabus for the instruction of calculus. While it is true that the textbook being used references numerous examples and applications of different mathematical software, typically no use of mathematical software is included in the course instruction.

By introducing mathematical software in the instruction of calculus the students can be exposed to some of the very powerful uses and applications of mathematical software. With the students' introduction to mathematical software in their calculus instruction via this package a major step is being taken in changing current methods of instruction. This package could be used at the United States Military Academy as well as other colleges and universities.

III. USE OF PACKAGE IN CALCULUS INSTRUCTION

This section addresses incorporating the integral package into the calculus instruction on two-dimensional integrals. The integral package is not intended to be used by itself to teach the calculus student how to find limits of integration and calculate two-dimensional integrals. Rather it should complement the classroom instruction and the homework problems on this topic.

The package should be introduced to the students after they have received the classroom instruction on two-dimensional integrals given in cartesian coordinates. The students should also first attempt to work a number of assigned homework problems that require finding the area of a region given in cartesian coordinates. After the students see the solutions to these problems, they could be shown the package.

The ideal way to introduce the package is in a computer lab in which all students can have hands-on use at a terminal. The instructor could either talk the students through turning on *Mathematica*, loading the package and using it or let the students simply follow the instructions in the tutorial found in Appendix B. At this point in the calculus instruction the students would only be ready for the first example problem in the tutorial. In this problem the students would use the

program to find the area of a region given in cartesian coordinates.

The instructor could have other problems for the students to solve. These could be problems given in cartesian coordinates which come directly out of the calculus textbook being used. The greater variety of problems which the students solve using the program will demonstrate the program's many capabilities to solve integral problems given in cartesian coordinates.

Once the students are shown how to use the integral program part of the package they should be introduced to the lab book which contains numerous worked problems. These problems show the students step-by-step how to find the limits of integration and evaluate the integral. These example problems will supplement the examples worked in the classroom instruction and those found in the calculus textbook being used. These detailed solutions will further reinforce the procedures established in the classroom for finding the limits of integration and setting up the proper integrals.

The students at this point would not have received any classroom instruction on integrals involving polar coordinates. As such the students should only be told of the program's ability to handle such integral problems. The students should first be taught how to graph in polar coordinates. Once the students have received this classroom instruction on graphing and on integrals in polar coordinates

and had the opportunity to work some of these problems they should be shown the polar coordinate capabilities of the integral program. For polar coordinates the students should be shown how the program can find the area between a polar curve and the origin or to find the area between two polar curves. Again the tutorial is best suited for this, with one example of each of these problems.

For all integral problems the student should first work the problems without the use of the program. The program should be used by the students to check their work and to identify any errors. All solutions given by the program include the plot of the functions and the region. The students can then see graphically presented the limits of integration.

It is important that the student always first attempt to do the problems on their own. The program is not intended to do the student's homework but rather it is to assist the student in learning. The goal is that the students can, by themselves, find the limits of integration regardless how a problem is presented. The program is an educational aid and should be used as such. This integral package properly incorporated into the calculus instruction can make learning to find limits of integration for two-dimensional integrals easier and fun. This integral package complements the classroom instruction on finding the limits of integration for two-dimensional integrals.

IV. WHY MATHEMATICA

While there are many types of mathematical software that I could have used, I selected **Mathematica**. Since I have not used any other mathematical software extensively I am unable to comment on the relative advantages or disadvantages of one software over another. One of my primary reasons for selecting **Mathematica** is its availability at both the Naval Postgraduate School and the United States Military Academy.

Mathematica also had a number of important attributes which lead me to select it. One such attribute of the software is its sophisticated graphing capabilities. Producing quality plots was essential to the package. **Mathematica** enabled me to use a variety of graphing capabilities and options. Not only did they plot the functions but also provided necessary information about the regions. This information was used to find limits of integration, determine upper and lower curves, left and right curves, or inner and outer curves, and finally the area of the region.

A second attribute of the software was its programming capability. **Mathematica** has an excellent pattern matching feature which was used extensively. I also found it easy to

program and debug in **Mathematica**. With Professor David Canright's expertise in **Mathematica** which was quite accessible, I felt confident in using it.

Another advantage or rather attribute of **Mathematica** is the ability for students to use the package and other **Mathematica** capabilities at the same time. In using **Mathematica** I was also able to take advantage of many of **Mathematica's** built-in functions.

Those platforms that support the program are IBM compatible personal computers with the Windows environment, the Unix workstation with X-Windows, and Macintosh computers. The program operation described herein is with the notebook interface and the Windows version of **Mathematica**. The details differ for the non-notebook interface found with the Unix version of **Mathematica**.

V. STRUCTURE OF THE PROGRAM

The **integral program** is organized into ten **Mathematica** packages which have nine major subprograms and numerous other defined functions. A package is simply a file containing **Mathematica** definitions. To access the functions defined in the packages, the packages must first be loaded. The file named **integral.ma** is at the core of the **integral program**. This file connects the ten files together along with the functions defined within them. When **integral.ma** is loaded it in turns loads all the other files which make up the program.

The program is started by calling the **subprogram graph**, found in file **integral.ma**. The rest of the program is built around **subprogram graph** which calls up the other subprograms. After the program is invoked **subprogram graph** prompts the user to respond to various instructions. It is in this manner in which different options are selected. Depending on which options are selected the appropriate subprograms are called (see Figure 2 on next page).

Program Architecture

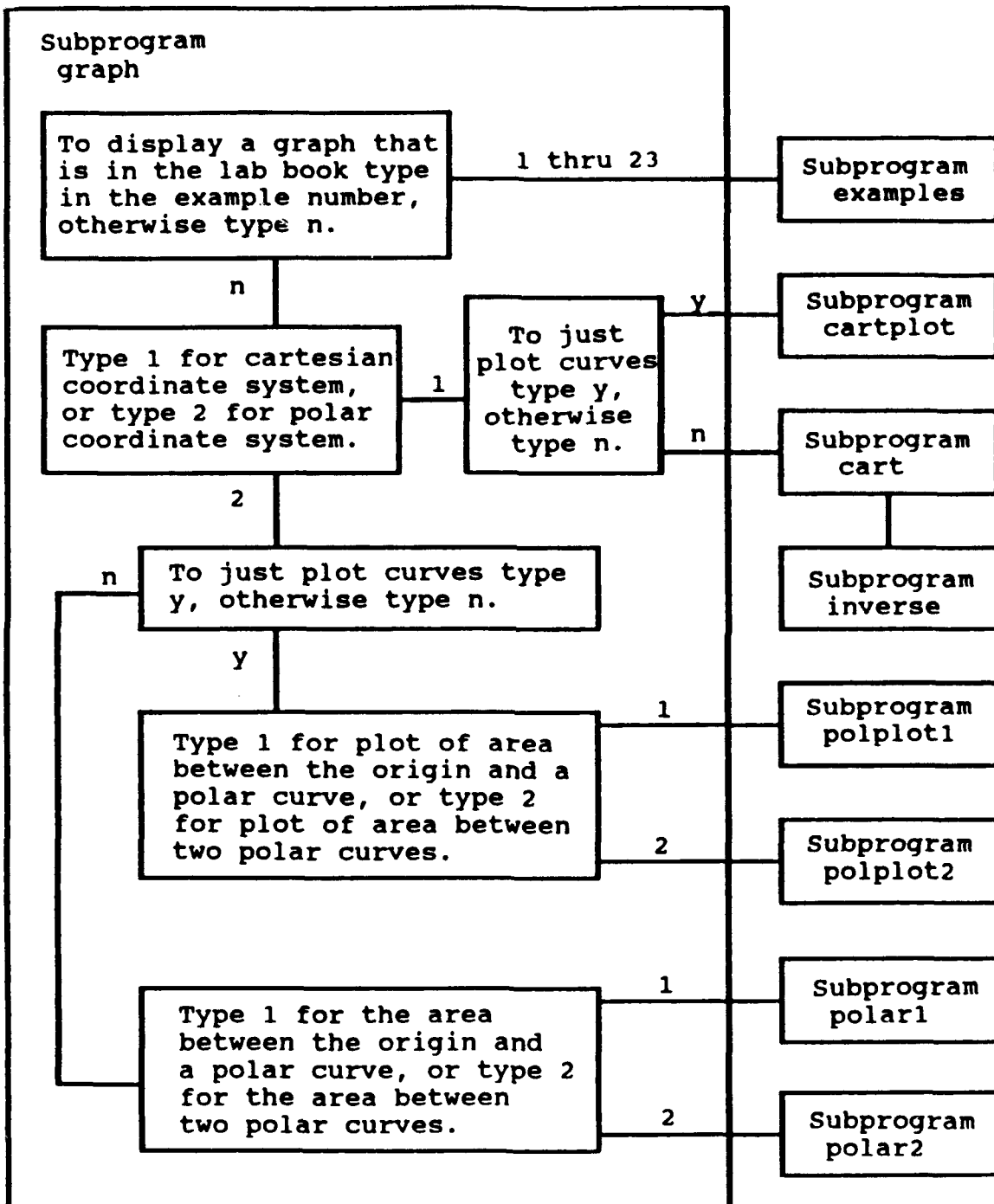


Figure 2

If the user desires to see the plot of the region for any of the example problems in Appendix B, **subprogram graph** calls **subprogram examples** (found in file **examples.ma**). If the user just wants to plot a set of curves given in cartesian coordinates **subprogram graph** calls up **subprogram cartplot** (found in file **cartplot.ma**). If the user has a cartesian coordinate problem to solve **subprogram graph** calls **subprogram cart** (found in file **cart.ma**). The regions in these types of problems must be given as functions of the independent variable x . The integration is then with respect to x . For problems given in cartesian coordinates **subprogram cart** calls up **subprogram inverse** (found in file **inverse.ma**). **Subprogram inverse** solves the problem by integrating over the region with respect to the variable y . **Subprogram inverse** first determines if integration with respect to the variable y is appropriate. If the functions are invertible the user is then asked if he desires to integrate the same problem with respect to the variable y .

If the user wants just a plot of the area between the origin and a polar curve, **subprogram graph** calls **subprogram polplot1** (found in file **polplot1.ma**). If the user wants just a plot of the area between two polar curves, **subprogram graph** calls **subprogram polplot2** (found in file **polplot2.ma**). If the problem is to find the area between a polar curve and the origin, **subprogram graph** calls **subprogram polar1** (found

in file `polar1.ma`). If the problem is to find the area between two polar curves, subprogram `graph` calls subprogram `polar2` (found in file `polar2.ma`).

Many of the example problems in Appendix B have regions which are shaded. File `shades.ma` contains the **Mathematica** definitions which produce the different shades for those example problems.

VI. SUBPROGRAM GRAPH

Subprogram graph is the main branch of the program. Depending upon the type of integral problem or option selected subprogram graph calls the appropriate subprograms.

After the program is started the first prompt says:

To display a graph that is in the lab book type in the example number, otherwise type n.

The user can type a number from 1 to 23 to call up subprogram examples. Subprogram examples displays the graph which corresponds to that number example problem found in the lab book.

If the user does not select this option the next prompt appears:

Type 1 for cartesian coordinate system, or type 2 for polar coordinate system.

If the user types 1 for cartesian coordinate system the following prompt appears:

To just plot curves type y, otherwise type n.

If the user types y, subprogram graph calls up subprogram cartplot which is the cartesian plot-only option. Otherwise if the user types n, subprogram graph calls up subprogram cart which solves those problems given in cartesian coordinates.

If when the user sees the prompt:

Type 1 for cartesian coordinate system, or type 2 for polar coordinate system.

If types 2 for polar coordinate system, the following prompt appears:

To just plot curves type y, otherwise type n.

If the user types y to select the polar plot-only option he then sees the next prompt:

Type 1 for plot of area between the origin and a polar curve, or type 2 for plot of area between two polar curves.

If the user types 1, subprogram graph calls up subprogram polplot1 which allows the user to graph the area between a polar curve and the origin. Otherwise if the user types 2, subprogram graph calls up subprogram polplot2 which allows the user to graph the area between two polar curves.

If the polar plot-only option is not selected the user sees the following prompt:

Type 1 for area between the origin and a polar curve, or type 2 for area between two polar curves.

If the user types 1, subprogram graph calls up subprogram polar1 which finds the area bounded by a polar curve and the origin. Otherwise if the user types 2, subprogram graph calls up subprogram polar2 which finds the area between two polar curves.

VII. SUBPROGRAM CART

Subprogram cart solves those problems given in cartesian coordinates. **Subprogram graph** calls up **subprogram cart** when the user indicates he wants to solve a problem given in cartesian coordinates. **Subprogram cart** can be used by itself without invoking the main program. In order to use this stand-alone option the user types **cart** and then activates the cell. The tutorial covers the many ways in which the cell can be activated.

Regardless how **subprogram cart** is called, the subprogram attempts to solicit all known information. The user is first told to enter the functions defining the region. These functions must be functions of the independent variable x . After the functions have been entered the user is told to indicate if any of the limits of integration is known. If the user indicates yes he is then prompted to input those known limits. The subprogram then includes the functions $x=(\text{left limit})$ and $x=(\text{right limit})$ as appropriate, to those functions of x already entered. If the left limit is not known and the right limit is either not known or greater than zero, the user is told to indicate if the region is restricted to the right half plane. If the right limit is not known and the left limit is either not known or less than zero, the user is told to indicate if the region is restricted to the left half

plane. In some problems the functions will intersect and form a region in the right half plane and one in the left half plane. The subprogram must know which region to consider. An example of this type of problem is the intersection of the functions $y=12-x^2$, $y=x$ and $y=0$ (see Figure 1 on page 5).

Once all this information is obtained, the subprogram is ready to solve the problem. It first obtains all the points of intersection for each of the curves and lines segments. For all pairs of equations it does this by simultaneously solving two equations with two unknowns, using **Mathematica's NSolve** function. The transcendental functions defined in the program are Taylor series expansions. When equations contain transcendental functions the subprogram uses these Taylor series expansions to find the approximate points of intersection. These approximations are then used as starting points in **Mathematica's FindRoots** function to find the points of intersection. **FindRoots** uses Newton's method to solve for these values.

Subprogram cart uses the x coordinates of the points of intersection, for each function entered by the user, to define the right and left limits for the curve. If these x coordinates are the same value the subprogram knows that the points of intersection, for the entered function, form a vertical line segment. In order for the program to solve a problem each curve must have just two endpoints.

If there are fewer than two endpoints, the functions do not bound a closed region. If there are more than two points of intersection, the functions intersect to form more than one region. In the latter case, the subprogram is unable to solve the problem correctly.

Consider the example where the lines $y=x$ and $y=5x$ are the only two functions entered. These functions intersect only at the point whose coordinates are $(0,0)$. The program then determines only one endpoint (i.e., the point $(0,0)$) for each function. These two functions by themselves, do not bound a closed region and the program does not have enough information.

Now consider the example where the functions are $y=x$, $y=0$ and $y=12-x^2$. The points of intersection, and thus the endpoints that the program determines, for the function $y=x$ are: $(-4,-4)$, $(0,0)$, $(3,3)$. Those for the function $y=0$ are: $(-3.464,0)$, $(0,0)$, $(3.464,0)$ while those for the function $y=12-x^2$ are: $(-4,-4)$, $(-3.464,0)$, $(3,3)$, $(3.464,0)$. Each of the functions has more than two endpoints and they intersect to form three separate regions (see Figure 1 on page 5).

If however the user indicates that the region is restricted to the right half plane the program discards those points of intersection which are in the left half plane. Similarly if the user indicates that the region is restricted to the left half plane the program discards those points of intersection which are in the right half plane.

Once the subprogram knows all the curves and their endpoints, **Mathematica's Plot** function samples the functions and determines the points which define each curve. Each curve and each line segment is then represented by a set of (x,y) points. While only the endpoints are needed here, the list of (x,y) points are formed to be used later in **subprogram inverse**. The x coordinates of the points of intersection for all the curves are taken to form a sorted list of points. Each adjacent pair of points in this list are the limits of integration for a subregion of the total region formed by the intersection of the functions.

Subprogram cart can find the area of a region bounded by two curves that intersect at two different points. In this case the user enters the two functions of the independent variable x and then indicates that no limits are known. An example of this type of problem is the region bounded by the two curves $y=x^3 + 1$ and $y=x^2 + x$ (see Figure 3 on page 27). **Subprogram cart** can find the area of the region bounded by two curves that intersect at one point and by a line segment perpendicular to the x -axis. In this case the user enters the two functions of the independent variable x , indicates that a limit is known and then enters that limit. An example of this type of problem is the region bounded by the two curves $y=x^2$ and $y=0$, which intersect at the origin, and the line $x=4$ (see Figure 4 on page 27). The third case is when both the right and left limits are known. Here the user enters the two

functions of the independent variable x , indicates that the limits are known and then inputs both limits. An example of this type of problem is when the two curves are $y=1$ and $y=1-x^2$ with the given limits $x=1$ and $x=4$ (see Figure 5 on page 28).

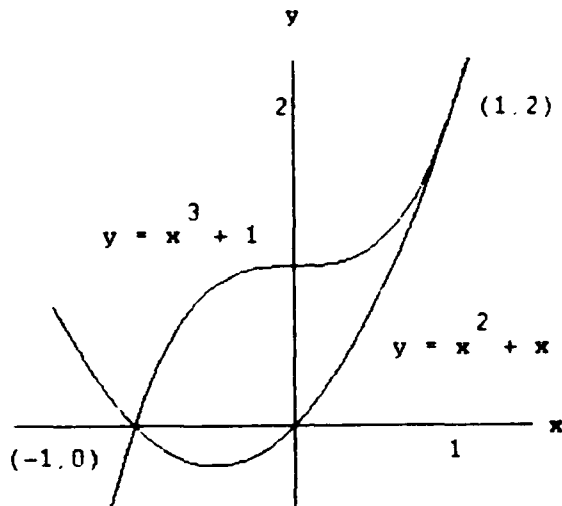


Figure 3

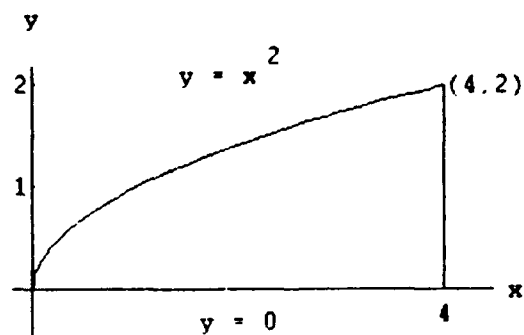


Figure 4

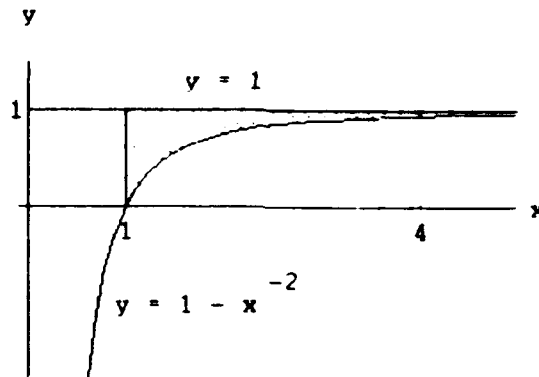


Figure 5

The maximum number of functions of x that the user can input is three. Then the maximum number of subregions which can be formed is two. Hence the program is able to solve for the area of a region which has changing boundaries. For example the functions $y=5x$, $y=x$ and $y=\cos x$ intersect to form a region which consists of two subregions (see Figure 6 below). The first subregion has $y=5x$ as its upper curve and $y=x$ as its lower curve. The second subregion has $y=\cos x$ as its upper curve and $y=x$ as its lower curve.

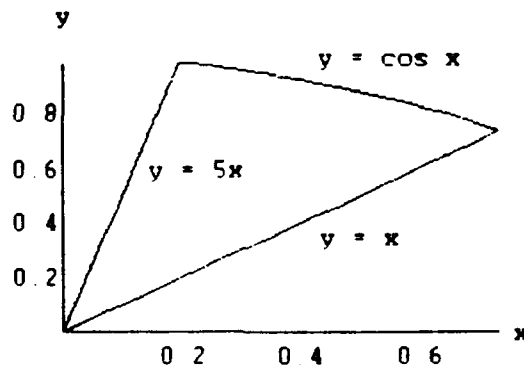


Figure 6

For each set of limit points and corresponding subregion, the subprogram determines the upper and lower curves. It does this by selecting those curves (now represented as lists of (x,y) points) whose largest and smallest x coordinate values bound the average value of that subregion's two limit points. The subprogram then uses Mathematica's `NIntegrate` function to integrate the upper curve minus the lower curve over the limits of that subregion. The region is then plotted. The output for each subregion includes: the limits of integration, the upper and lower curves, and the area. The area of the total region is also provided.

In order to solve a problem, the subprogram `cart` must be able to identify that region. It was mentioned previously that the subprogram has limitations if the functions intersect to form more than one region. The region of interest may be such that the program can not directly solve the problem. An example of this type of problem is again when the functions are $y=x$, $y=0$ and $y=12-x^2$ (see Figure 1, page 5). Suppose that the region of interest is that region which lies partly in the left half plane and partly in the right half plane. For this type of problem the user would have to plot the functions using the plot-only option (`subprogram cartplot`). When the user indicates that his problem is given in cartesian coordinates he is given the option to just plot the functions. If this option is selected the functions are plotted and the points where the curves intersect are provided

as output. From the plot of the functions and the points of intersection the user could determine the limits for the region of interest. The user could then use this information and the program to solve the problem.

In the example problem above the plot (see Figure 1, page 5) shows that the upper curve for the subregion in the left half plane is $y=12-x^2$ while the lower curve is $y=0$. For the subregion in the right half plane the upper curve is $y=12-x^2$ and the lower curve is $y=x$. The user could treat each subregion separately and use the program to solve for the areas individually and then add the two areas.

The subprogram has no difficulty in solving problems in which the functions bound a single region which lies, either in part or whole, in the left half plane. Thus while subprogram `cart` has limitations the plot-only option can be used in conjunction with it to solve certain of these types of problems.

VIII. SUBPROGRAM INVERSE

Subprogram inverse integrates those problems given in cartesian coordinates with respect to the variable y . First, however, the subprogram determines if it is appropriate to integrate a problem with respect to y . It attempts to find the inverse of each function of x that was input. If not all the inverse functions exist **subprogram inverse** stops execution. If on the other hand they all exist the user is asked if he wants to integrate the same problem with respect to y . **Subprogram inverse** uses Mathematica's `InverseFunction` to find the inverse functions.

Subprogram inverse works similar to **subprogram cart** to solve the problem. The y -values of the points of intersection for the curves (found in **subprogram cart**), are limits of integration. There can exist limits, however, that are not any of the y -values of the points of intersection. An example of this type of problem is when the functions are $y=12-x^2$ and $y=-3+x+x^2$ (see Figure 7 on page 32). These functions intersect at the points $(-3,3)$ and $(2.5,5.75)$. The first function is the upper curve and takes on a maximum y -value of 12. The second function is the lower curve and takes on a minimum y -value of -3.25 . Hence there are three subregions. The limits of integration for the first region are $y=-3.25$ and $y=3$. The limits of integration for the second region are $y=3$

and $y=5.75$. The limits of integration for the third region are $y=5.75$ and $y=12$. The y -values -3.25 and 12 are limits while they are not any of the y -values for the points of intersection.

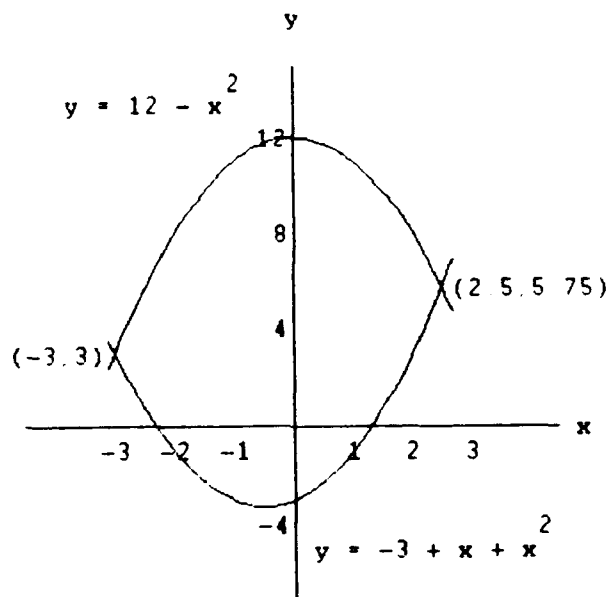


Figure 7

The subprogram finds the minimum and maximum values of the functions and includes them as limit points. In subprogram **cart** all the curves are represented by lists of points. From these lists subprogram **inverse** selects the minimum and maximum values which y takes on in the curves. Once all the limits of integration are known they are placed in a sorted list. Each adjacent pair of points are then limits of integration of a subregion of the total region.

For each subregion the subprogram determines the right and left curves in the same manner in which subprogram **cart** finds the upper and lower curves. Some inverse functions have a

positive branch and a negative branch. These functions are identified as having a square root term involving the variable x . Subprogram **inverse** checks to see if either the right curve or left curve for a subregion has this type of inverse function. If an inverse function is of this type, the subprogram must determine whether it is the positive or the negative branch.

If it is the negative branch, the subprogram calls routine **negbranch**, which transforms the inverse function. The term in the inverse function involving the square root is then replaced with the negative of that term. For example consider the region bounded by the functions $y=0$, $y=-4+x^2$ and $x=-4$ (see Figure 8 on page 34). The inverse of $y=-4+x^2$ is $x=\sqrt{y+4}$. When the region is integrated with respect to y the right curve is $x=\sqrt{y+4}$ and the left curve is $x=-4$. But the right curve is a negative branch so we have $x=-\sqrt{y+4}$ as the right curve. In the calculation of the integral the integrand is the right curve minus the left curve, or $(-\sqrt{y+4}) - (-4) = -\sqrt{y+4}+4$.

If the right curve for a subregion is the same as the left curve, the subprogram calls another routine **bothbranches**. This routine works similar to the routine **negbranch**. It insures that the region integrated over is the positive branch minus the negative branch (or twice the positive branch). For

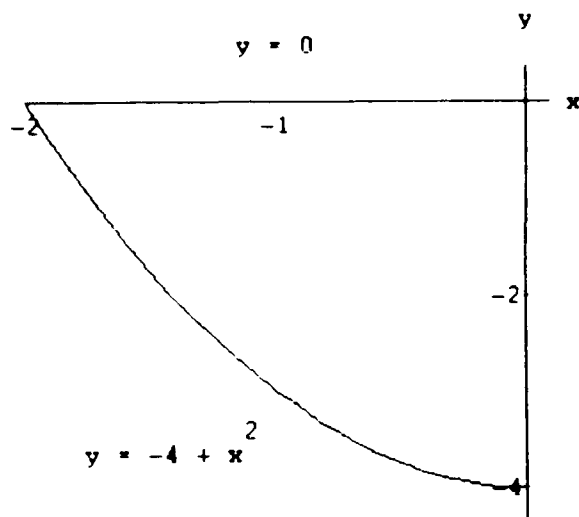


Figure 8

example consider the region bounded by the functions $y=x^2$ and $y=4$ (see Figure 9 on page 35). The inverse function of $y=x^2$ is $x=\sqrt{y}$. But in this problem the right curve is the positive branch and the left curve is the negative branch. Therefore the integrand is $\sqrt{x} - (-\sqrt{x}) = 2\sqrt{x}$, which is the right curve minus the left curve.

The area of the subregions are then found using Mathematica's `NIntegrate` function. The output for each subregion includes: the limits of integration, the left and right curves, and the area. The area of the total region is also provided.

Subprogram inverse as mentioned previously works only when all inverse functions exist. Also stated was that it only integrates with respect to the variable y when it is appropriate to do so. There are functions whose inverse exists but which are complicated functions to integrate

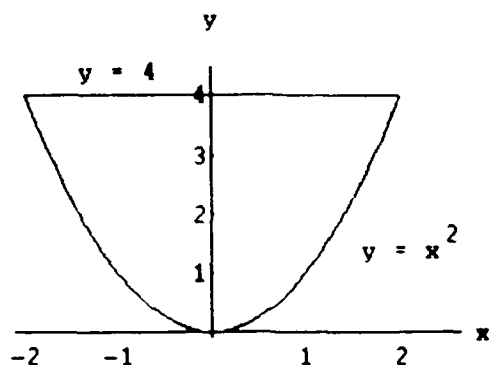


Figure 9

(Mathematica has no problem but the calculus student might). For those problems involving transcendental functions the subprogram will not integrate with respect to y , even if such integration is possible. Integration with respect to the variable y is supposed to save the calculus student time. Usually the inverse of a function containing transcendental functions is complicated and more difficult to integrate.

IX. SUBPROGRAM CARTPLOT

Subprogram cartplot is an option which allows the user to graph functions given in cartesian coordinates. After the user indicates that a problem is given in cartesian coordinates, **subprogram graph** asks if the user just wants to plot the functions. If the user selects this option **subprogram graph** calls up **subprogram cartplot**. The user first enters all functions of the independent variable x and then those lines perpendicular to the x -axis. The user enters the ranges for both the independent variable x , and the dependent variable y . The functions are then plotted and the points where the curves intersect are given.

This plot-only option can be used to estimate the coordinates of where the functions intersect and thus the limits of integration. There is no limit on the number of functions which can be plotted on one graph with this option. A single function can be plotted if desired. **Subprogram cartplot** uses **Mathematica's Plot** function to plot all graphs.

Subprogram cartplot can be used by itself without invoking the main program. This is valuable in saving time if the user just wants to plot several graphs. In order to use this stand-alone option the user types **cartplot** and then activates the cell. The tutorial covers the many ways in which the cell can be activated.

Regardless how subprogram cartplot is called the first prompt the user sees is:

To just plot curves type y, otherwise type n.

Suppose the user wants to plot the functions $y=x$, $y=0$ and $y=12-x^2$. The user types **y** to select the plot-only option. The next message the user sees is:

Enter the curves, one at a time. When finished type n.

y =

The user then types **x** and presses Enter to input the first function. The above message appears again and the user then types **0** and presses Enter to input the second function. Once again the above message appears and the user inputs the last function by typing $12-x^2$ and pressing Enter. The above message appears again and this time the user types **n** and presses Enter to indicate that he is finished.

The user then sees the message:

Enter lines perpendicular to the x-axis. When finished type n.

Since there are none the user types **n**. If there were lines to enter they would have been entered one at a time as the functions of the independent variable x were entered above.

The user then sees the message:

Enter lower limit for x-range of plot.

The user must guess what this value is as well as the upper limit for the x -range and both the lower and upper limits for the y -range. Suppose the user knows these values. For

example, to enter the lower limit for the x-range he then types -5 and presses Enter. The user then sees the message:

Enter upper limit for x-range of plot.

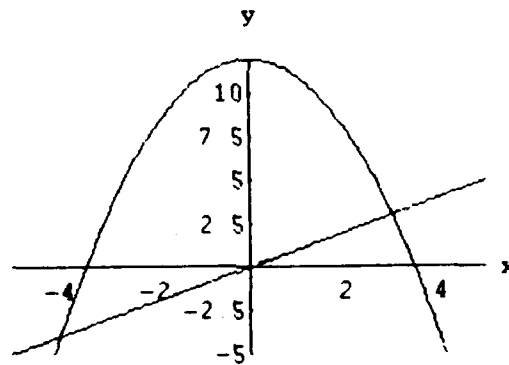
He types 5 and presses Enter and sees the message:

Enter lower limit for y-range of plot.

He types -5 and presses Enter and sees the message:

Enter upper limit for y-range of plot.

He types 12 and the plot is displayed along with the points where the curves intersect (see Figure 10 below). After the plot is displayed the user may want to change the x or y range of the plot. To do this the user must call up **subprogram cartplot** again and enter all the information.



The points where the curves intersect are:

$\{(-4., -4.), (-3.464, 0.), (0., 0.), (3., 3.), (3.464, 0)\}$

Figure 10

X. SUBPROGRAM POLAR1

Subprogram `polar1` finds the area bounded by a polar curve and the origin. Subprogram `graph` calls up subprogram `polar1` when the user indicates the problem is to find the area between the origin and a polar curve. Subprogram `polar1` can be used by itself without invoking the main program. In order to use this stand-alone option the user types `polar1` and then activates the cell.

Regardless how subprogram `polar` is called, the user first enters the polar curve r , which bounds the region. The polar curve must be a function of the independent variable Q representing the polar angle θ . The subprogram asks the user if the limits of integration are known and then to enter them, if they are known. For those problems in which the limits are known in advance the subprogram then graphs the region and solves the problem. The subprogram uses `Mathematica's PolarPlot` function to produce all plots. The output includes the limits of integration, the polar curve and the area bounded by the curve.

A polar curve may be negative for certain ranges of the independent variable Q . For example the polar curve $r=\sin 3Q$ takes on negative values in the range $Q=\pi/3$ to $Q=2\pi/3$. A polar curve may also be undefined for certain ranges of the independent variable Q (i.e., when equal to the square root of

a negative number). The polar curve $r = \sqrt{\cos 2Q}$ is defined only for Q between $Q = -\pi/4$ and $Q = \pi/4$ and for Q between $Q = 3\pi/4$ and $Q = 5\pi/4$ (see Figure 11 below).

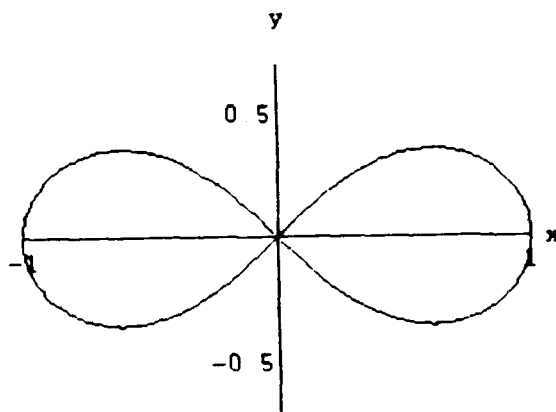


Figure 11

The subprogram determines for which values of Q the function r is equal to zero. It uses Mathematica's `NSolve` function to do this. For all regions between these values of Q (where $r=0$) and the values of Q entered as limits, the function r must be positive, negative, or undefined. The subprogram determines the case for each such region.

When the function r is defined for a region the subprogram uses Mathematica's `NIntegrate` function to find the area of that region. The limits of integration are those values of Q which define the region. Since the problem is to find the

area between the polar curve and the origin the absolute value of the evaluated integral is taken. Therefore a region where the polar curve r is negative contributes positive area to the total area between the curve and the origin. If the function r is not defined over a region, no area is contributed.

If the user does not know the limits of integration the polar curve is plotted for $Q=0$ to $Q=2\pi$. The plot of the curve is displayed along with those Q values for which the function r is equal to zero. After a brief delay the user is told that the polar curve is plotted from $Q=0$ to $Q=2\pi$ and is asked to input the limits of integration. From this point on the subprogram works exactly like it did above when the limits of integration were known in advance.

If the area bounded by a polar curve and the origin is made up of more than one region, the limits of integration for all those regions are provided as output. For example the area between the polar curve $r=\sqrt{\cos 2Q}$ and the origin for $Q=0$ to $Q=3\pi/2$ consists of two separate regions (see Figure 12 on page 42). The limits of integration for the first region are $a_1=0$. and $b_1=\pi/4$ while the limits of integration for the second region are $a_2=3\pi/4$ and $b_2=5\pi/4$. These limits are displayed along with the graph of the region and the total area.

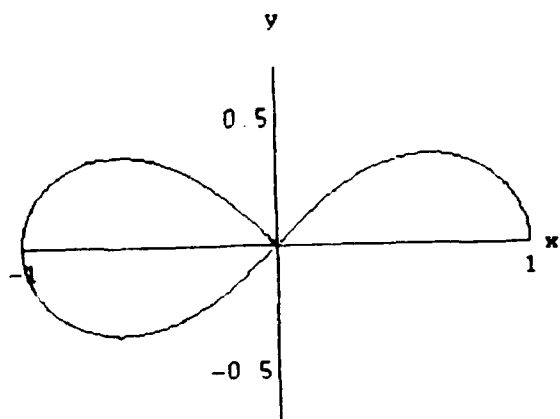


Figure 12

The subprogram does not tell how much area is contributed by each region. If the user wants to know this, he could treat each region as a separate problem. The program could then be used to determine the areas individually, one at a time.

XI. SUBPROGRAM POLPLOT1

Subprogram polplot1 is an option which allows the user to graph the area between the origin and a polar curve. After the user indicates that a problem is given in polar coordinates, **subprogram graph** asks if the user just wants to plot the functions. If the user selects this plot-only option the program then asks the user to select the type of plot. The choices are: (1) the plot of the area between the origin and a polar curve, and (2) the plot of the area between two polar curves. When the user selects (1), **subprogram graph** calls up **subprogram polplot1**.

The user enters the polar curve which bounds the region and then indicates if the limits are known. If the limits are not known, the polar curve is plotted for the polar angle $Q=0$ to $Q=2\pi$. Those points where the polar curve is equal to zero are displayed along with the graph. The user is then given an option to change the limits. If the user selects this option the polar curve is then replotted for the new limits of the polar angle Q , which the user enters. **Subprogram polplot1** uses **Mathematica's PolarPlot** function to plot the graphs.

Subprogram polplot1 can be used by itself without invoking the main program. In order to use this stand-alone option the user types **polplot1** and then activates the cell.

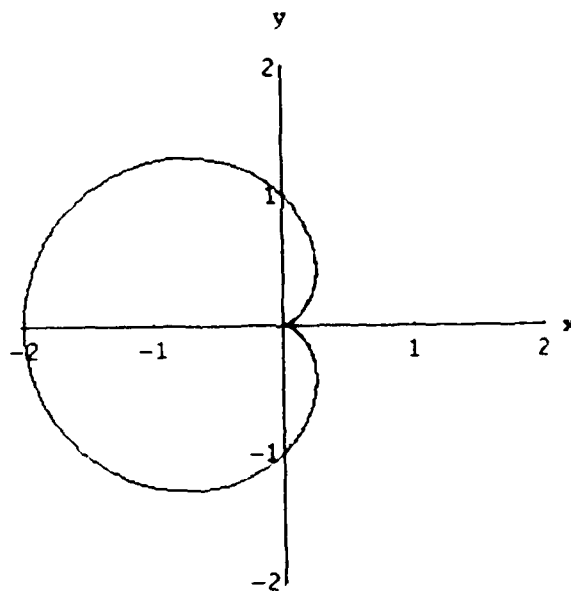
Regardless how subprogram polplot1 is called the first prompt the user sees is:

Enter the polar curve which bounds the region.

If the user wanted to plot the polar curve $r = 1 - \cos Q$, he would type $1 - \cos[Q]$ and press Enter. The next message the user would see is:

If the limits are known type y, otherwise type n.

Suppose the user types n. The polar curve is then displayed along with those points where the polar curve is equal to zero (see Figure 13 below).



The polar curve r is equal to zero when Q is: $(0, 2\text{ Pi})$

Figure 13

After a brief delay the user sees the message:

The polar curve is plotted from $Q=0$ to $Q=2\pi$. If you want to change the limits type y , otherwise type n .

Now suppose the user wants to change the lower limit to $Q=\pi/2$ and the upper limit to $Q=3\pi/2$. He types y and the next message that appears is:

Input new lower limit.

The user types $\pi/2$ and presses Enter. The next message that appears is:

Input new upper limit.

The user types $3\pi/2$ and presses Enter. The plot of the polar curve is then displayed for the new range of the polar angle Q (see Figure 14 below).

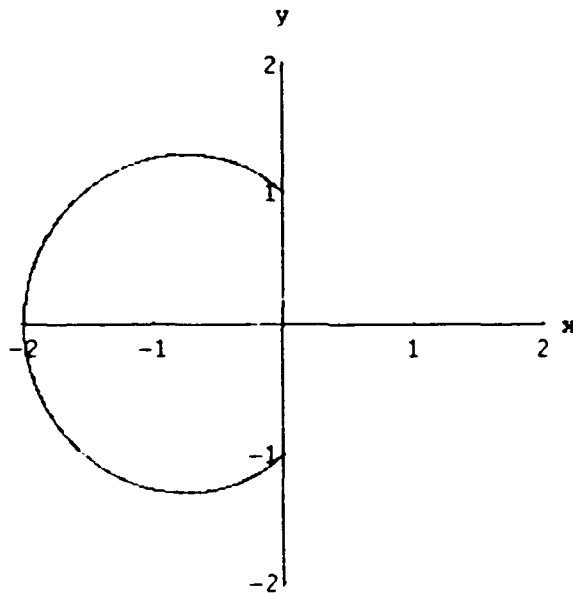


Figure 14

XII. SUBPROGRAM POLAR2

Subprogram `polar2` finds the area between two polar curves. Subprogram `graph` calls up subprogram `polar2` when the user indicates the problem is to find the area bounded between two polar curves. Subprogram `polar2` can be used by itself without invoking the main program. In order to use this stand-alone option the user types `polar2` and then activates the cell.

Regardless how subprogram `polar2` is called, the subprogram directs the user to enter the two polar curves. These polar curves must be functions of the independent variable Q . After the two polar curves have been entered the user is asked if the limits of integration are known and to input them if known.

The program determines those values of Q where the curves intersect. The functions which define the curves are represented as Taylor Series expansions. The subprogram uses `Mathematica's NRoots` function to obtain the approximate values of Q for which the curves intersect. These values are then used as starting points in `Mathematica's FindRoot` function. This function uses Newton's Method to find the points of intersection.

If the limits are not known the subprogram then plots the two polar curves for $Q=0$ to $Q=2\pi$. All plots are produced using `Mathematica's PolarPlot` function. The values of Q for

which the two polar curves intersect are also provided. After a brief delay the user is told that the curves are plotted for $Q=0$ to $Q=2\pi$. The subprogram asks the user if he wants to change these limits. If the user desires he can then input the new limits. If the limits of integration are known and inputted initially, the program plots the two polar curves over this region.

Once the limits are entered the subprogram calls up a number of routines which check for certain types of conditions. Routine **negcheck** determines if the polar curves are negative. The user is told if one or more of the curves are negative in the range $Q=\text{lower limit}$ to $Q=\text{upper limit}$. The user is also told that the program does not handle this type of problem. Routine **complexcheck** determines the ranges of Q for which the curves are undefined (if they are). Routine **diffcheck** checks to see if the outer and inner polar curve changes between the upper and lower limits. If the outer curve and inner curve changes the subprogram lets the user know this. The user is told that a point of intersection cannot lie between the two limits. The user is then directed to input new limits.

Once all conditions have been checked and the program determines that the problem is well defined it calculates the area between the two polar curves. The subprogram has two routines which it uses to find the area. If both curves are defined over the entire interval routine **findarea2** is used.

This routine uses Mathematica's `NIntegrate` function to evaluate the single integral. If however the inner curve is undefined over certain ranges of Q the routine `findarea1` is used to find the area.

An example is when the outer curve is the circle $r=1$ and the inner curve is $r=\sqrt{\cos 2Q}$ (see Figure 15 below). Here the inner curve is not defined for the polar angle $Q=\pi/4$ to $Q=3\pi/4$ and for $Q=-3\pi/4$ to $-\pi/4$. Routine `findarea1` first finds the area between the outer polar curve and the origin and then finds the area between the inner polar curve and the origin. The difference between these two areas is the area between the two polar curves.

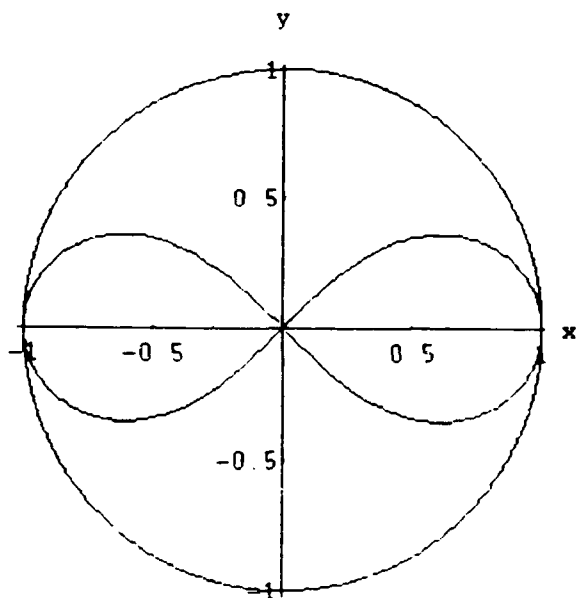


Figure 15

The region bounded by the two polar curves is then plotted. The output includes the limits of integration, the outer curve, the inner curve, and the total area of the region. The program does not provide the areas of any subregions. If the user wanted to know this, each subregion could be treated as a separate problem. The program could then be used to find the area of each individual subregion, one at a time.

XIII. SUBPROGRAM POLPLOT2

subprogram polplot2 is an option which allows the user to graph the area between two polar curves. After the user indicates that a problem is given in polar coordinates, **subprogram graph** asks if the user just wants to plot the functions. If the user selects this plot-only option the program then asks the user to select the type of plot. The choices are: (1) the plot of the area between the origin and a polar curve, and (2) the plot of the area between two polar curves. When the user selects (2), **subprogram graph** calls up **subprogram polplot2**.

The user enters the two polar curves which bound the region and then indicates if the limits are known. If the limits are not known, the polar curves are plotted for the polar angle $Q=0$ to $Q=2\pi$. Those points where the two polar curves intersect are displayed along with the graph. The user is then given an option to change the limits. If the user selects this option the polar curves are then replotted for the new limits of the polar angle Q , which the user enters. **Subprogram polplot2** uses **Mathematica's PolarPlot** function to plot the graphs.

Subprogram polplot2 can be used by itself without invoking the main program. In order to use this stand-alone option the user types **polplot2** and then activates the cell.

Regardless how subprogram polplot2 is called the first prompt the user sees is:

Enter one of the polar curves which bounds the region.

Suppose the user wanted to plot the area between the two polar curves $r=2$ and $r=1+\sin Q$. To enter the first polar curve the user would type 2 and press Enter. The next message the user would see is:

Enter the other polar curves which bounds the region.

The user would type $1+\sin[Q]$ and press Enter to enter the second polar curve. The next message the user would see is:

If the limits are known type y, otherwise type n.

Suppose the user types n. The polar curves are then displayed along with those points where the polar curves intersect (see Figure 16 on page 52).

After a brief delay the user sees the message:

The polar curves are plotted from $Q=0$ to $Q=2\pi$. If you want to change the limits type y, otherwise type n.

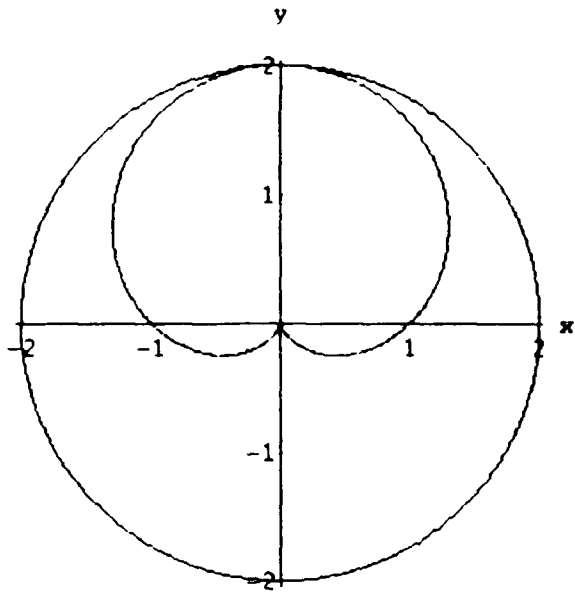
Now suppose the user wants to change the upper limit to $Q=\pi$. He types y and the next message that appears is:

Input new lower limit.

The user types 0 and presses Enter. The next message that appears is:

Input new upper limit.

The user types pi and presses Enter. The plot of the two polar curves is then displayed for the new range of the polar angle Q (see Figure 17 on page 52).



The two polar curves intersect at $Q = \frac{\pi}{2}$

Figure 16

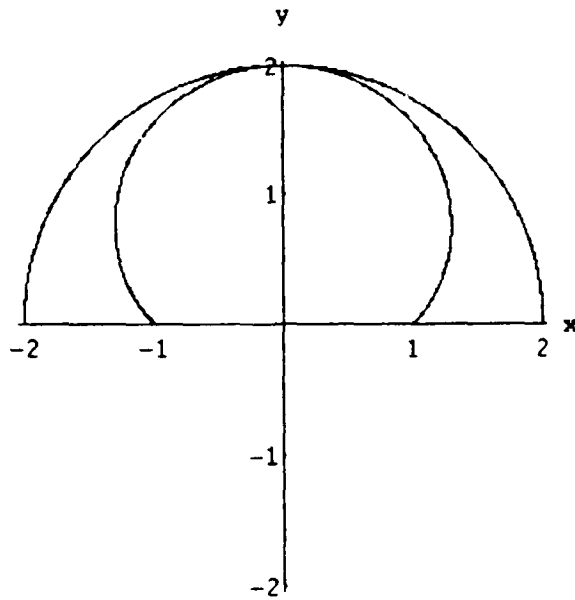


Figure 17

XIV. SUBPROGRAM EXAMPLES

The file named **examples.ma** contains the **Mathematica** code for each of the graphs found in the lab book. All of the graphs are in a package and can be displayed by the **subprogram examples**. When in the main program the first prompt says:

To display a graph that is in the lab book type the example number, otherwise type n.

The user can then type a number from 1 to 23 to call up **subprogram examples**. The subprogram then displays the graph which corresponds to that number example problem.

There may be only limited value in displaying two-dimensional graphs which are already in the lab book. However, when this integral package is expanded to include the three-dimensional case this could be a very useful option. **Mathematica** has a sophisticated three-dimensional graphing capability which uses a three-color lighting system to illuminate three dimensional surfaces. **Mathematica** also has the ability to change the point of view interactively to explore three-dimensional graphics. For these types of problems being able to display the graphs becomes important.

XV. FUNCTIONS AND NOTATION

A. FUNCTIONS

When inputting information both the program and **Mathematica** are case sensitive. Some transcendental functions are defined within the program code and must be in lower case letters. These transcendental functions are represented as Taylor Series expansions. They were needed to solve algebraically for where two curves intersect, using **NRoots**. These approximate points were then used as starting points for **Mathematica's FindRoot** function which uses Newton's Method.

Mathematica has its own built-in transcendental functions which in most instances are spelled exactly the same. The only difference is that all **Mathematica** functions begin with an upper case letter.

Below are those functions which the program recognizes:

sin[x]	sec[x]
cos[x]	exp[x]
tan[x]	ln[x]
cot[x]	sqrt[x]
csc[x]	

B. NOTATION

1. Brackets and Parentheses

Square brackets and parentheses are intended for different purposes. Square brackets are used for specifying

arguments of functions. Parentheses are used for grouping. Without parentheses, multiplication and division have a higher precedence than addition and subtraction.

2. Mathematical Symbols

The standard mathematical operations are referred to with symbols. Those symbols used are:

<u>Mathematical Symbol</u>	<u>Operation</u>
+	plus, add
-	minus, subtract
*	times, multiply
/	divide
^	power

3. Other Symbols

The symbol Q is used for theta in problems given in polar coordinates. All trigonometric functions must be functions of the variable Q . The program does not recognize any other such variable. The symbol π represents the numerical value of the mathematical constant π .

XVI. FUTURE WORK

Right now the program just handles two-dimensional problems. The package could be extended to include three-dimensional problems. The three-dimensional case would include problems given in cartesian as well as cylindrical and spherical coordinate. Three-dimensional problems could also be included in the lab book. Subprogram examples then could display the graphs for these example problems. Mathematica's representation of three-dimensional plots would give the student a feel for the depth of the region and insight into how the surfaces intersect.

Currently there is no flexibility in defining the dependent and independent variables for either cartesian or polar coordinate problems. In cartesian coordinates, functions are of the independent variable x and the dependent variable is y . In polar coordinates the polar function r must always be a function of the independent variable θ . The program could be improved by allowing the user to determine the variable names (i.e., the user may want the dependent variable to be t).

The focus of all future work should support the intent of the integral package, which is to teach the calculus student how to find the limits of integration for integral problems.

As such, all future work should be to improve upon the package's ability to do this.

APPENDIX A

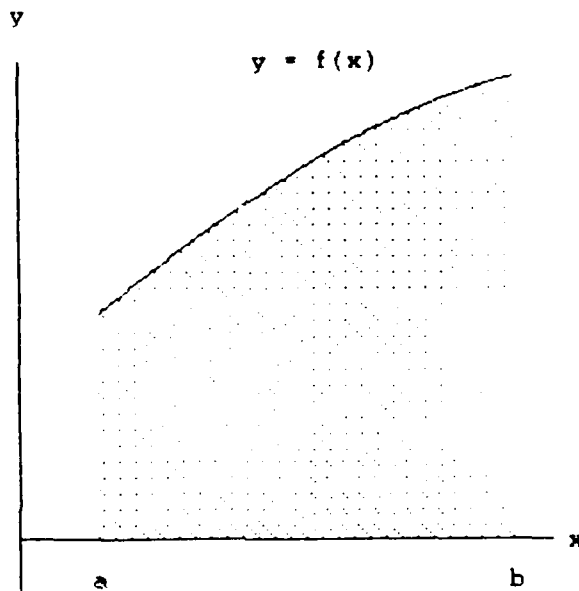
LAB BOOK

I. INTRODUCTION

One of the most difficult topics in calculus is deciding on the limits of integration. This lab book shows the calculus student how a given domain is plotted and teaches the student how to find the limits of integration when evaluating two-dimensional integrals.

A. Integration

Integral calculus is the mathematics we use to find lengths, areas, and volumes of irregular shapes and to calculate the average values of functions. The development of integral calculus starts from the calculation of areas.



We can use the integral calculus to find the areas of regions like the shaded one here.

B. Lab Book Organisation

This package uses numerous example problems for the two dimensional case. It considers cartesian as well as polar coordinates. Almost all of the problems begin with a description of a given domain over which an integral is to be evaluated. The given domain is plotted and the way in which the limits of integration are found is shown. Finally the integrals are set up and evaluated.

II. CARTESIAN COORDINATE SYSTEM

A. Finding the Area under the Graph of a Nonnegative Continuous Function.

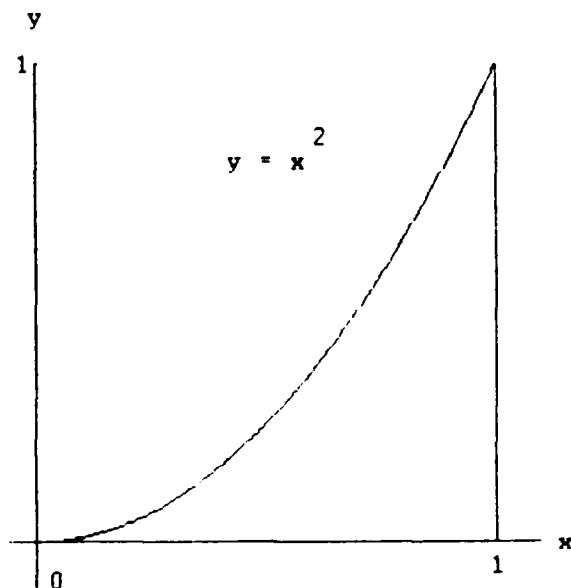
How to Find the Area under the Graph of a Non-negative Continuous Function $y=f(x)$ from a to b .

STEP 1: Find an antiderivative $F(x)$ of $f(x)$.

STEP 2: Calculate $F(b)-F(a)$. This number will be the area under the curve from a to b .

Example 1

Find the area under the curve $y=x^2$ from $x=0$ to $x=1$.



Solution. In this example $f(x)=x^2$, $a=0$, and $b=1$. We start with a graph of the function and identify the area of the region that lies between the curve and the x -axis from a to b . We then find the area in two steps.

STEP 1: Find an antiderivative $F(x)$ of $f(x)=x^2$.

$$F(x) = \frac{x^3}{3}$$

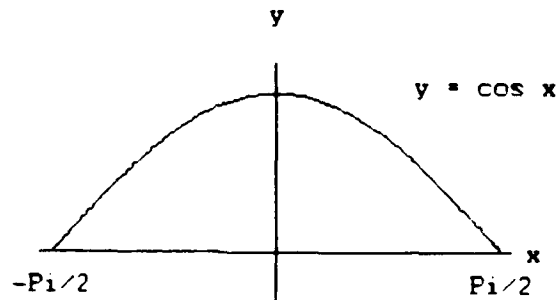
STEP 2: Calculate $F(1)-F(0)$.

$$F(1)-F(0) = \frac{(1)^3}{3} - \frac{(0)^3}{3} = \frac{1}{3} - 0 = \frac{1}{3}$$

The area is $1/3$.

Example 2

Find the area under one arch of the curve $y=\cos x$.



Solution. In this example $f(x)=\cos x$, however a and b are not explicitly given. We graph the function and observe that it crosses the x -axis at $x=-\pi/2$ and $x=\pi/2$. Note that $\cos(-\pi/2) = 0$ and $\cos(\pi/2) = 0$. Thus we have $a=-\pi/2$ and $b=\pi/2$. We then find the area in two steps.

STEP 1: Find an antiderivative $F(x)$ of $f(x)=\cos x$.

$$F(x)=\sin x$$

STEP 2: Calculate $F(-\pi/2)-F(\pi/2)$.

$$F(-\pi/2)-F(\pi/2)=\sin(-\pi/2)-\sin(\pi/2)=1-(-1)=2$$

The area is 2.

B. The Average Value of a Function

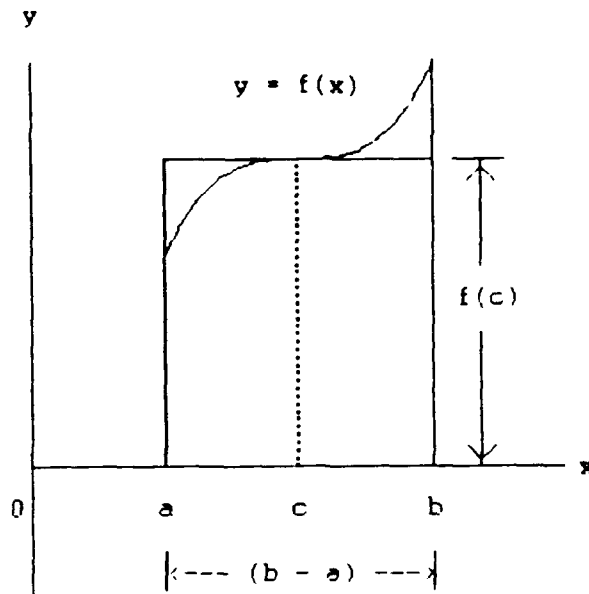
The average value of the function f on $[a,b]$ is the integral of f divided by the length of the interval.

Average value of f on $[a,b]$ is $\frac{1}{b-a} \int_a^b f(x)dx$.

If f is continuous and nonnegative on $[a,b]$, its average value is the height of a rectangle whose area,

$$f(c)(b-a) = \int_a^b f(x)dx,$$

is the area under the graph of f from a to b .



Example 3

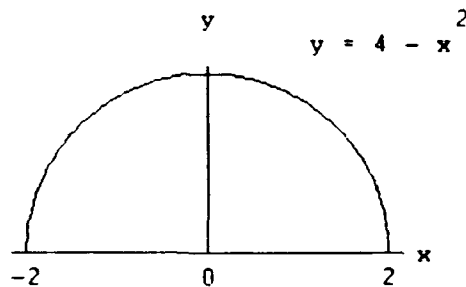
Find the average value of $f(x) = \sqrt{4 - x^2}$, on the interval $[-2, 2]$.

Solution: We graph the integrand $f(x)$ over the interval of integration $[-2, 2]$ and see that the graph is a semicircle of radius 2. The area between the semicircle and the x -axis is

$$\text{Area} = \frac{1}{2} \pi r^2 = \frac{1}{2} \pi (2)^2 = 2\pi.$$

Because the area is also the value of the integral of f from $x=-2$ to $x=2$, the average value of f on $[-2, 2]$ is

$$\text{Average value} = \frac{1}{2 - (-2)} \int_{-2}^2 \sqrt{4 - x^2} dx = \frac{1}{4} 2\pi = \frac{\pi}{2}.$$

**Example 4**

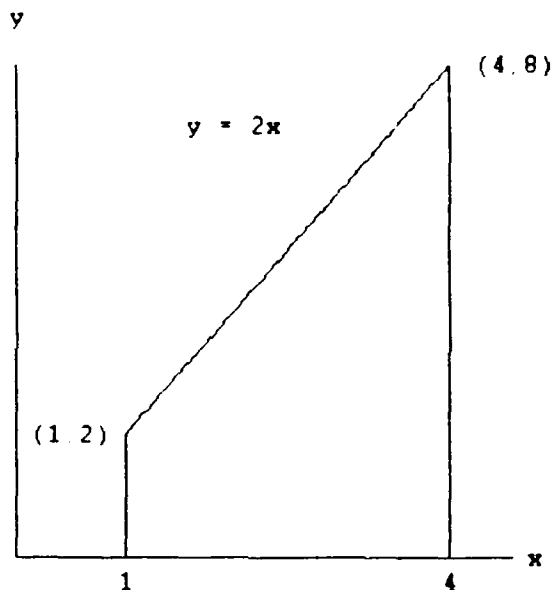
Find the average value of $f(x) = 2x$ on the interval $[1, 4]$.

Solution. We graph the integrand $f(x)$ over the interval of integration $[1, 4]$ and we see that the graph of the area is a trapezoid whose base is on the x -axis. The area of a trapezoid is

$$\text{Area} = \frac{1}{2} \text{ base}(h_1 + h_2) = \frac{1}{2} (3)(8 + 2) = 15.$$

Because the area is also the value of the integral of f from $x=1$ to $x=4$, the average value of f on $[1, 4]$ is

$$\text{Average value} = \frac{1}{4-1} \int_1^4 2x dx = \frac{1}{3} (15) = 5.$$



C. Area Between Curves

This section shows how to find the area of a region in the coordinate plane by integrating the functions that define the region's boundaries.

Definition

If functions f and g are continuous and if $f(x)$ is greater than or equal to $g(x)$ throughout the interval $[a, b]$ such that x is in $[a, b]$, then the area of the region between the curves $y=f(x)$ and $y=g(x)$ from a to b is the integral of $(f-g)$ from a to b :

$$\text{Area} = \int_a^b [f(x) - g(x)] dx.$$

How to find the Area between Two Curves

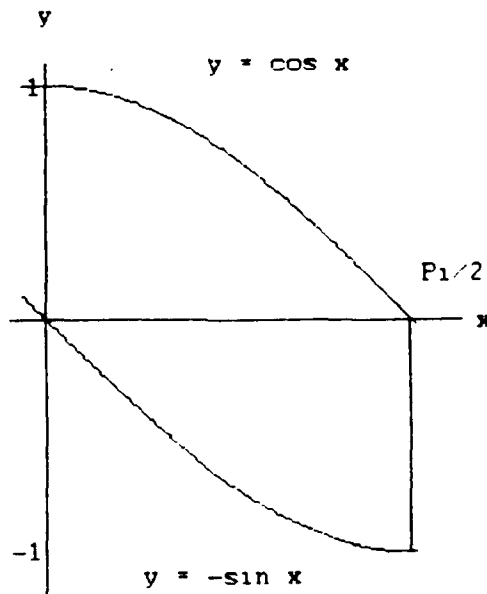
1. Graph the curves together. This tells you which is f , (upper curve) and which is g , (lower curve). It also helps to find the limits of integration if you do not already know them.
2. Find the limits of integration.
3. Write a formula for $f(x) - g(x)$. Simplify it if you can.
4. Integrate $f(x) - g(x)$ from a to b . The number you get is the area.

Example 5

Find the area between the curve $y = \cos x$ and the curve $y = -\sin x$ from $x = 0$ to $x = \pi/2$.

Solution.

STEP 1: The graphs. We graph the curves together. The upper curve is $y = \cos x$, so we take $f(x) = \cos x$ in the area formula. The lower curve is $y = -\sin x$, so $g(x) = -\sin x$.



STEP 2: The limits of integration. They are given:

$$a=0 \text{ and } b=\pi/2.$$

STEP 3: The formula for $f(x)-g(x)$. From Step 1,

$$\begin{aligned} f(x)-g(x) &= \cos x - (-\sin x) \\ &= \cos x + \sin x \end{aligned}$$

STEP 4: Integrate $f(x)-g(x)$ from $a=0$ to $b=\pi/2$.

$$\begin{aligned} \int_0^{\pi/2} (\cos x + \sin x) dx &= [\sin x - \cos x]_0^{\pi/2} \\ &= [(1-0) - (0-1)] = 2 \end{aligned}$$

The area between the curves is 2.

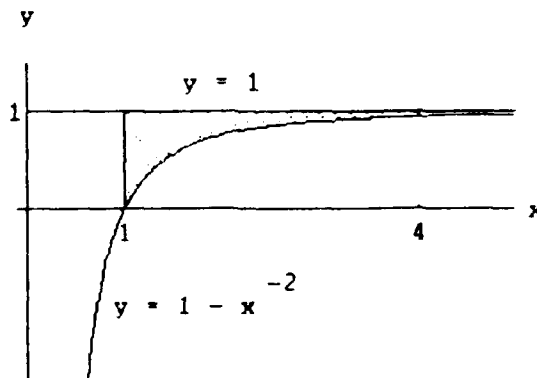
Example 6

Find the area between the curves $y=1$ and $y=1-x^{-2}$ for $x=1$ to $x=4$.

Solution.

STEP 1: The graphs. We graph the curves together. The upper curve is $y=1$, so $f(x)=1$. The lower curve

is $y=1-x^{-2}$, so $g(x)=1-x^{-2}$.



STEP 2: The limits of integration. The limits of integration are given: $a=1$ and $b=4$.

STEP 3: The formula for $f(x)-g(x)$.

$$f(x)-g(x)=1-(1-x^2)=x^2$$

STEP 4: Integrate.

$$\text{Area} = \int_a^b [f(x)-g(x)] dx = \int_1^4 x^2 dx = - \left. \frac{1}{x} \right|_1^4 = - \left(\frac{1}{4} - \frac{1}{1} \right) = \frac{3}{4}$$

The area of the region is $3/4$.

D. Curves that Cross

When a region is determined by curves that cross, the crossing points give the limits of integration.

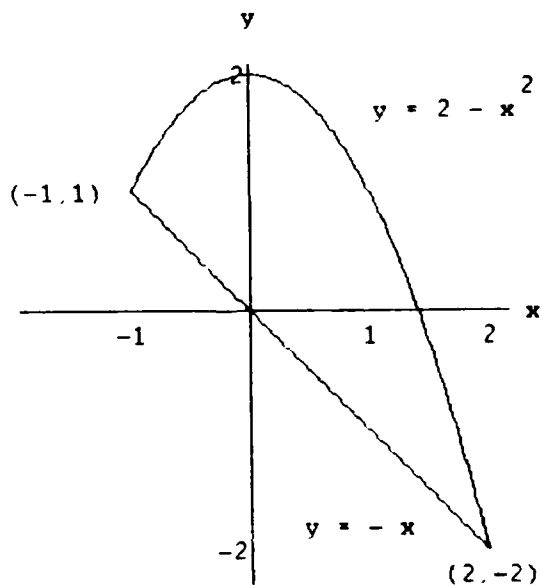
Example 7

Find the area of the region enclosed by the parabola $y=2-x^2$ and the line $y=-x$.

Solution.

STEP 1: The graphs. We graph the curves together.

The upper curve is $y=2-x^2$, so $f(x)=2-x^2$. The lower curve is $y=-x$, so $g(x)=-x$. The x -coordinates of the points where the parabola and line cross are the limits of integration. We find them in Step 2.



STEP 2: The limits of integration. We find the limits of integration by solving the equations $y=2-x^2$ and $y=-x$ simultaneously for x :

$$\begin{aligned} 2-x^2 &= -x && \text{(Equate } f(x) \text{ and } g(x)) \\ x^2-x-2 &= 0 && \text{(Transpose)} \\ (x+1)(x-2) &= 0 && \text{(Factor)} \\ x &= -1, x=2. && \text{(Solve)} \end{aligned}$$

The region runs from $x=-1$ on the left to $x=2$ on the right. The limits of integration are $a=-1$ and $b=2$.

STEP 3: The formula for $f(x)-g(x)$.

$$f(x)-g(x) = (2-x^2) - (-x) = 2+x-x^2$$

STEP 4: Integrate.

$$\text{Area} = \int_a^b [f(x)-g(x)] dx = \int_{-1}^2 (2+x-x^2) dx$$

$$= \left| 2x + \frac{x^2}{2} - \frac{x^3}{3} \right|_{-1}^2$$

$$= \left(4 + \frac{4}{2} - \frac{8}{3} \right) - \left(-2 + \frac{1}{2} - \frac{1}{3} \right)$$

$$= 6 + \frac{3}{2} - \frac{9}{3} = \frac{9}{2}$$

The area of the region is $9/2$.

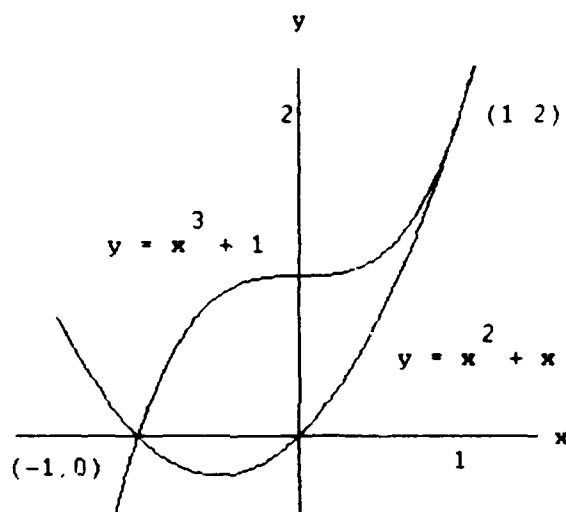
Example 8

Find the area of the region enclosed by the curve $y=x^3+1$ and the curve $y=x^2+x$.

Solution.

STEP 1: The graphs. We graph the curves together.

The upper curve is $y=x^3+1$, so $f(x)=x^3+1$. The lower curve is $y=x^2+x$, so $g(x)=x^2+x$. The x-coordinates of the points where the curves cross are the limits of integration. We find them in Step 2.



STEP 2: The limits of integration. We find the limits by solving the equations $y=x^3+1$ and $y=x^2+x$ simultaneously for x :

$$x^3+1 = x^2+x \quad (\text{Equate } f(x) \text{ and } g(x))$$

$$x^3-x^2-x+1 = 0 \quad (\text{Transpose})$$

$$(x+1)(x-1)^2 = 0 \quad (\text{Factor})$$

$$x=-1, x = 1. \quad (\text{Solve})$$

The region runs from $x=-1$ on the left to $x=1$ on the right. The limits of integration are $a=-1$ and $b=1$.

STEP 3: The formula for $f(x) - g(x)$.

$$f(x) - g(x) = x^3 + 1 - (x^2 + x) = x^3 - x^2 - x + 1$$

STEP 4: Integrate.

$$\text{Area} = \int_a^b [f(x) - g(x)] dx = \int_{-1}^1 [x^3 - x^2 - x + 1] dx$$

$$= \left. \left[\frac{1}{4} x^4 - \frac{1}{3} x^3 - \frac{1}{2} x^2 + 1 \right] \right|_{-1}^1$$

$$= \left. \left[\frac{1}{4} (1)^4 - \frac{1}{3} (1)^3 - \frac{1}{2} (1)^2 + 1 \right] \right| - \left. \left[\frac{1}{4} (-1)^4 - \frac{1}{3} (-1)^3 - \frac{1}{2} (-1)^2 + 1 \right] \right|$$

$$= \left. \left[\frac{1}{4} - \frac{1}{3} - \frac{1}{2} + 1 \right] \right| - \left. \left[\frac{1}{4} + \frac{1}{3} - \frac{1}{2} + 1 \right] \right| = \frac{4}{3}$$

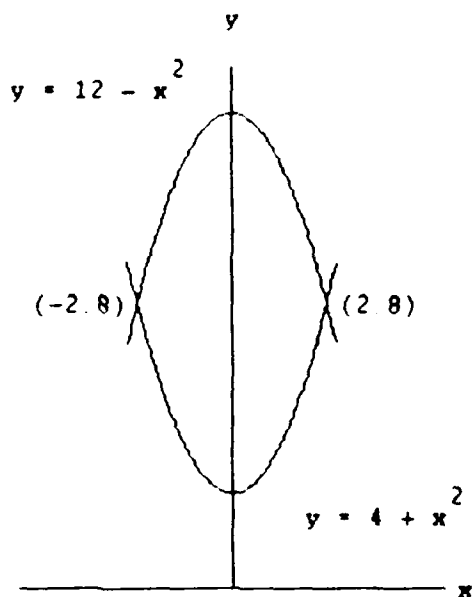
The area of the region is $4/3$.

Example 9

Find the area of the region enclosed by the parabola $y = 12 - x^2$ and the parabola $y = 4 + x^2$.

Solution.

STEP 1: The graphs. We graph the curves together. The upper curve is $y = 12 - x^2$, so $f(x) = 12 - x^2$. The lower curve is $y = 4 + x^2$, so $g(x) = 4 + x^2$. The x -coordinates of the points where the two parabolas cross are the limits of integration. We find them in Step 2.



STEP 2: The limits of integration. We find the limits by solving the equations $y=12-x^2$ and $y=4+x^2$ simultaneously for x :

$$\begin{aligned}
 4+x^2 &= 12-x^2 && \text{(Equate } f(x) \text{ and } g(x)) \\
 2x^2 &= 8 && \text{(Transpose)} \\
 x^2 &= 4 && \text{(Divide by 2)} \\
 x=2, x=-2. &&& \text{(Solve)}
 \end{aligned}$$

The region runs from $x=-2$ on the left to $x=2$ on the right. The limits of integration are $a=-2$ and $b=2$.

STEP 3: The formula for $f(x)-g(x)$.

$$f(x)-g(x)=12-x^2-(4+x^2)=8-2x^2$$

STEP 4: Integrate.

$$\text{Area} = \int_a^b [f(x)-g(x)] dx = \int_{-2}^2 [8-2x^2] dx$$

$$= \left[8x - \frac{2}{3} x^3 \right]_{-2}^2$$

$$= [8(2) - \frac{2}{3}(2)^3] - [8(-2) - \frac{2}{3}(-2)^3] = \frac{62}{3}$$

The area of the region is $62/3$.

B. Boundaries with Changing Formulas

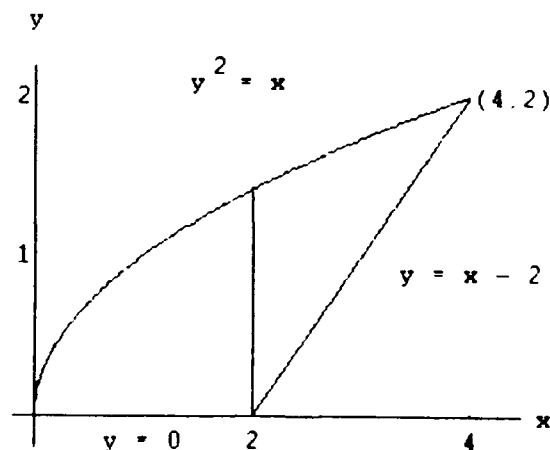
If the formula for one of the bounding curves changes at some point across the region, you may have to add two or more integrals to find the area.

Example 10

Find the area of the region in the first quadrant bounded above by the curve $y = \sqrt{x}$ and bounded below by the x-axis and the line $y = x - 2$.

Solution.

STEP 1: The graphs. We graph the curves together. The entire upper boundary of the region consists of the curve $y = \sqrt{x}$, so $f(x) = \sqrt{x}$. The lower boundary consists of two curves, first $y = 0$ for $x = 0$ to $x = 2$ and then $y = x - 2$ for $x = 2$ to $x = 4$. Hence the formula for $g(x)$ changes from $g(x) = 0$ for $x = 0$ to $x = 2$, to $g(x) = x - 2$ for $x = 2$ to $x = 4$.



STEP 2: The limits of integration. The limits of integration for the pair $f(x)=\sqrt{x}$ and $g(x)=0$ are $a=0$ and $b=2$. For the pair $f(x)=\sqrt{x}$ and $g(x)=x-2$, the left-hand limit is $a=2$ and the right-hand limit is the x -coordinate of the upper point where the line crosses the parabola. To find it, we solve the equations $y=\sqrt{x}$ and $y=x-2$ simultaneously for x :

$$\begin{aligned} \sqrt{x} &= x-2 && \text{(Equate } f(x) \text{ and } g(x)) \\ x &= (x-2)^2 = x^2 - 4x + 4 && \text{(Square)} \\ x^2 - 5x + 4 &= 0 && \text{(Transpose)} \\ (x-1)(x-4) &= 0 && \text{(Factor)} \\ x &= 1, x=4. && \text{(Solve)} \end{aligned}$$

The value $x=1$ does not satisfy the equation $\sqrt{x}=x-2$. It is an extraneous root introduced by squaring. The $x=4$ gives our upper limit of integration.

STEP 3: The formulas for $f(x)-g(x)$.

$$\text{For } x=0 \text{ to } x=2: f(x)-g(x)=\sqrt{x}-0=\sqrt{x},$$

$$\begin{aligned} \text{For } x=2 \text{ to } x=4: f(x)-g(x) &= \sqrt{x}-(x-2) \\ &= \sqrt{x}-x+2. \end{aligned}$$

STEP 4: Integrate. We have two integrals to evaluate. Their sum is the area.

$$\begin{aligned} \text{Area} &= \int_a^b [f(x)-g(x)] dx \\ &= \int_0^2 \sqrt{x} dx + \int_2^4 [\sqrt{x}-x+2] dx \\ &= \left. \frac{2}{3} x^{3/2} \right|_0^2 + \left. \left(\frac{2}{3} x^{3/2} - \frac{x^2}{2} + 2x \right) \right|_2^4 \end{aligned}$$

$$= \frac{2}{3} (2)^{3/2} + \left| \frac{2}{3} (4)^{3/2} - \frac{(4)^2}{2} + 2(4) \right| - \left| \frac{2}{3} (2)^{3/2} - \frac{(2)^2}{2} + 2(2) \right|$$

$$= \frac{10}{3}$$

The area of the region is $10/3$.

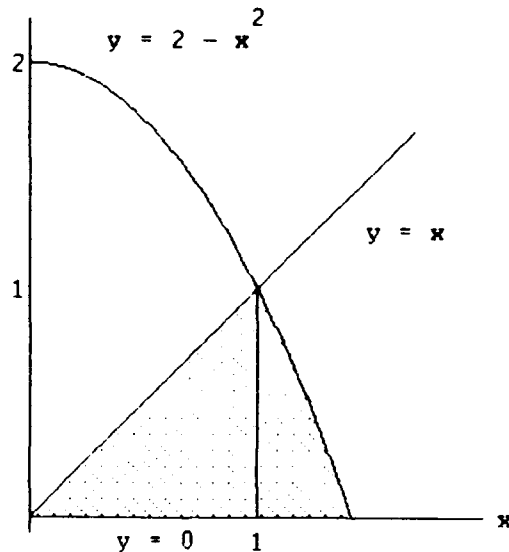
Example 11

Find the area of the region in the first quadrant bounded above by the curves $y=x$ and $y=2-x^2$ and below by the x -axis.

Solution.

STEP 1: The graphs. We graph the curves together. The upper boundary consists of two curves, first

$y=x$ for $x=0$ to $x=1$ and then $y=2-x^2$ for $x=1$ to $x=\sqrt{2}$. Hence the formula for $f(x)$ changes from $f(x)=x$ for $x=0$ to $x=1$, to $f(x)=2-x^2$ for $x=1$ to $x=\sqrt{2}$. The entire lower boundary of the region consists of the curve $y=0$, so $g(x)=0$.



STEP 2: The limits of integration. For the pair $f(x)=x$ and $g(x)=0$ the left-hand limit is $a=0$. The right-hand limit is the x -coordinate of the upper point where the line crosses the parabola. To find it, we solve the equations $y=x$ and $y=2-x^2$ simultaneously for x :

$$\begin{aligned}
 x &= 2 - x^2 && \text{(Equate } f(x) \text{ and } g(x)) \\
 x^2 + x - 2 &= 0 && \text{(Transpose)} \\
 (x+2)(x-1) &= 0 && \text{(Factor)} \\
 x &= -2, x = 1. && \text{(Solve)}
 \end{aligned}$$

Since x is greater than or equal to zero in the first quadrant we must pick $x=1$. The limits of integration are $a=0$ and $b=1$. For the pair $f(x)=2-x^2$ and $g(x)=0$ the left-hand limit is 1 and the right-hand limit is obtained by solving for where the curve $y=2-x^2$ crosses the x -axis:

$$\begin{aligned}
 2 - x^2 &= 0 && \text{(Equate } f(x) \text{ and } g(x)) \\
 2 &= x^2 && \text{(Transpose)} \\
 x &= \sqrt{2}. && \text{(Solve)}
 \end{aligned}$$

This region runs from 1 on the left to $\sqrt{2}$ on the right. The limits of integration are $a=1$ and $b=\sqrt{2}$.

STEP 3: The formulas for $f(x)-g(x)$.

For $x=0$ to $x=1$: $f(x)-g(x)=x-0=x$,

For $x=1$ to $x=\sqrt{2}$: $f(x)-g(x)=2-x^2-0=2-x^2$.

STEP 4: Integrate. We have two integrals to evaluate. Their sum is the area.

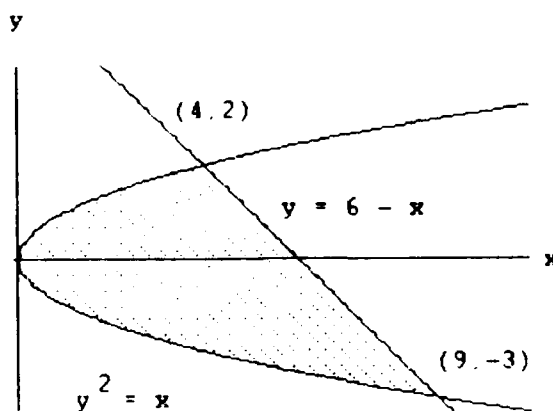
$$\begin{aligned}
 \text{Area} &= \int_a^b [f(x)-g(x)] dx = \int_0^1 x dx + \int_1^{\sqrt{2}} [2-x^2] dx \\
 &= \left. \frac{x^2}{2} \right|_0^1 + \left. \left(2x - \frac{x^3}{3} \right) \right|_1^{\sqrt{2}} \\
 &= \frac{(1)^2}{2} + [2\sqrt{2} - \frac{(\sqrt{2})^3}{3}] - [2(1) - \frac{(1)^3}{3}] \\
 &= .719
 \end{aligned}$$

Example 12

Find the area bounded by the parabola $y^2 = x$ and the line $y = 6 - x$.

Solution.

STEP 1: The graphs. We graph the curves together. The upper boundary consists of two curves, first $y = \sqrt{x}$ for $x = 0$ to $x = 4$ and then $y = 6 - x$ for $x = 4$ to $x = 9$. Hence the formula for $f(x)$ changes from $f(x) = \sqrt{x}$ for $x = 0$ to $x = 4$ to $f(x) = 6 - x$ for $x = 4$ to $x = 9$. The entire lower boundary of the region consists of $y = -\sqrt{x}$, so $g(x) = -\sqrt{x}$.



STEP 2: The limits of integration. For the pair $f(x) = \sqrt{x}$ and $g(x) = -\sqrt{x}$ the left-hand limit is 0. The right-hand limit is the x -coordinant of the upper point where the line crosses the parabola. To find it, we solve the equations $y = \sqrt{x}$ and $y = 6 - x$ simultaneously for x :

$$\sqrt{x} = 6 - x \quad (\text{Equate } f(x) \text{ and } g(x))$$

$$x = x^2 - 12x + 36 \quad (\text{Square})$$

$$x^2 - 13x + 36 = 0 \quad (\text{Transpose})$$

$$(x - 4)(x - 9) = 0 \quad (\text{Factor})$$

$$x = 4, \quad x = 9. \quad (\text{Solve})$$

The upper point where the parabola and line cross is at $x=4$. The limits of integration are $a=0$ and $b=4$. For the pair $f(x)=6-x$ and $g(x)=-\sqrt{x}$ the left-hand limit is $x=4$ and the right-hand limit is the x -coordinate of the lower point where the parabola and line cross. We found this point when we solved the two equations above simultaneously for x . The right-hand limit is $x=9$. The limits of integration are $a=4$ and $b=9$.

STEP 3: The formulas for $f(x)-g(x)$.

$$\begin{aligned} \text{For } x=0 \text{ to } x=4: \quad f(x)-g(x) &= \sqrt{x} - (-\sqrt{x}) \\ &= 2\sqrt{x}, \end{aligned}$$

$$\begin{aligned} \text{For } x=4 \text{ to } x=9: \quad f(x)-g(x) &= 6-x - (-\sqrt{x}) \\ &= 6-x + \sqrt{x}. \end{aligned}$$

STEP 4: Integrate. We have two integrals to evaluate. Their sum is the area.

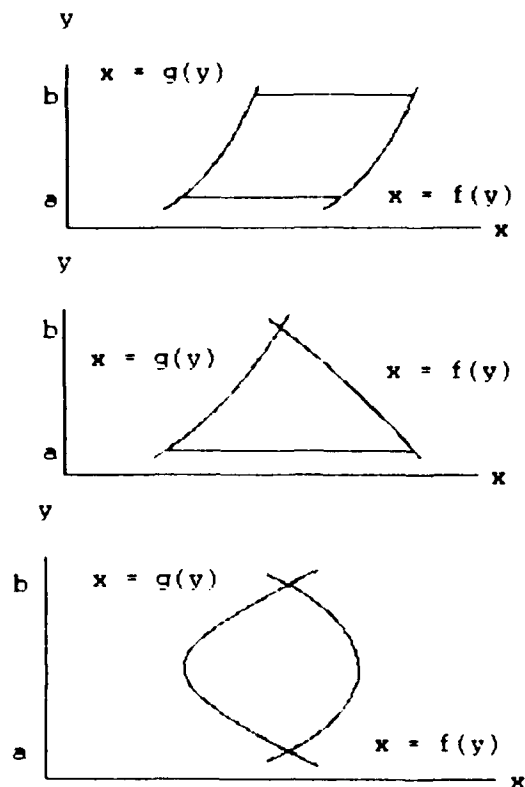
$$\begin{aligned} \text{Area} &= \int_a^b [f(x)-g(x)] dx = \int_0^4 2\sqrt{x} dx + \int_4^9 [6-x+\sqrt{x}] dx \\ &= \left| \frac{4}{3} x^{3/2} \right|_0^4 + \left| 6x - \frac{x^2}{2} + \frac{2}{3} x^{3/2} \right|_4^9 \\ &= \frac{4}{3} (4)^{3/2} + \left[6(9) - \frac{(9)^2}{2} + \frac{2}{3} (9)^{3/2} \right] - \left[6(4) - \frac{(4)^2}{2} + \frac{2}{3} (4)^{3/2} \right] \\ &= 20.833 \end{aligned}$$

The area of the region is 20.833.

F. Integrating with Respect to y

When a region's bounding curves are described by giving x as a function of y , the basic formula changes.

For regions like these,



use the formula:

$$\text{Area} = \int_a^b [f(y) - g(y)] dy.$$

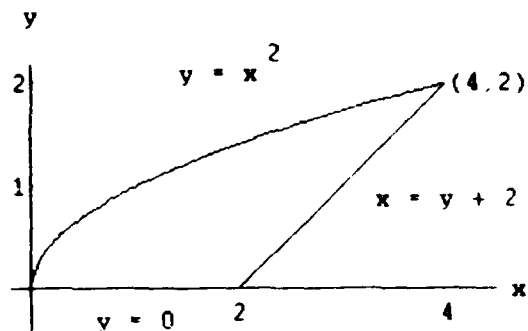
The only difference is that we are now integrating with respect to y instead of x . We can sometimes save time by doing so. The basic steps are the same as before.

Example 13

Find the area of the region between the curves $x=y^2$ and $x=y+2$ in the first quadrant.

Solution.

STEP 1: The graphs. We graph the curves together. The right-hand curve is $x=y+2$, so $f(y)=y+2$. The left-hand curve is $x=y^2$, so $g(y)=y^2$.



STEP 2: The limits of integration. The lower limit of integration is $y=0$. The upper limit is the y -coordinate of the upper point where the line crosses the parabola. We find it by solving the equations $x=y+2$ and $x=y^2$ simultaneously for y :

$$y+2=y^2 \quad (\text{Equate } f(y) \text{ and } g(y))$$

$$y^2+y+2=0 \quad (\text{Transpose})$$

$$(y+1)(y-2)=0 \quad (\text{Factor})$$

$$y=-1, y=2. \quad (\text{Solve})$$

The upper limit of integration is 2. (The value $y=-1$ gives the point of intersection below the x -axis.)

STEP 3: The formula for $f(y)-g(y)$.

$$f(y)-g(y)=y+2-y^2$$

STEP 4: Integrate.

$$\begin{aligned} \text{Area} &= \int_a^b [f(y)-g(y)] dy = \int_0^2 [2+y-y^2] dy \\ &= \left[2y + \frac{y^2}{2} - \frac{y^3}{3} \right]_0^2 = 4 + \frac{4}{2} - \frac{8}{3} = \frac{10}{3} \end{aligned}$$

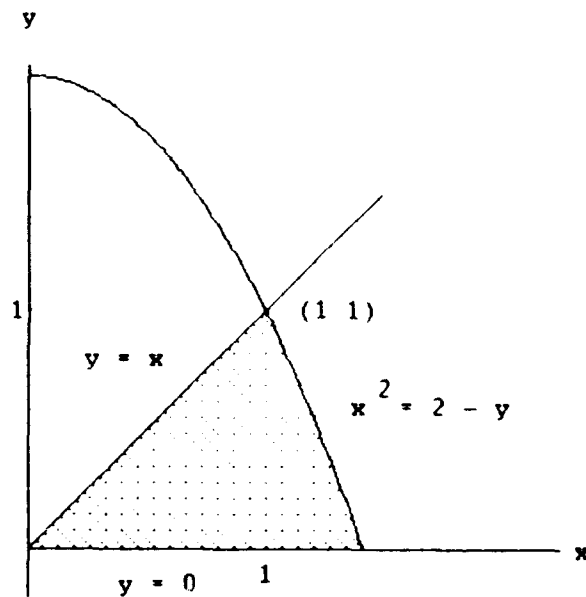
The area of the region is $10/3$.

Example 14

Find the area of the region between the curves $x=y$ and $x=\sqrt{2-y}$ in the first quadrant.

Solution.

STEP 1: The graphs. We graph the curves together. The right-hand curve is $x=\sqrt{2-y}$, so $f(y)=\sqrt{2-y}$. The left-hand curve is $x=y$, so $g(y)=y$.



STEP 2: The limits of integration. The lower limit of integration is $y=0$. The upper limit is the y-coordinate of the upper point where the line crosses the parabola. We find it by solving the equations $x=y$ and $x=\sqrt{2-y}$ simultaneously for y :

$$y = \sqrt{2-y} \quad (\text{Equate } f(y) - g(y))$$

$$y^2 = 2 - y \quad (\text{Square})$$

$$y^2 + y - 2 = 0 \quad (\text{Transpose})$$

$$(y+2)(y-1) = 0 \quad (\text{Factor})$$

$$y = -2, y = 1. \quad (\text{Solve})$$

The upper limit of integration is 1. (The value $y=-1$ gives the point of intersection below the x-axis.) The limits of integration are $a=0$ and $b=1$.

STEP 3: The formula for $f(y) - g(y)$.

$$f(y) - g(y) = \sqrt{2-y} - y$$

STEP 4: Integrate.

$$\text{Area} = \int_a^b [f(y) - g(y)] dy = \int_0^1 [\sqrt{2-y} - y] dy$$

$$= \left| -\frac{2}{3}(2-y)^{3/2} - \frac{y^2}{2} \right|_0^1$$

$$= \left[-\frac{2}{3}(2-1) - \frac{(1)^2}{2} \right] = -\frac{2}{3} - \frac{1}{2} + \frac{2}{3} - \frac{3}{2} = 0 = .719$$

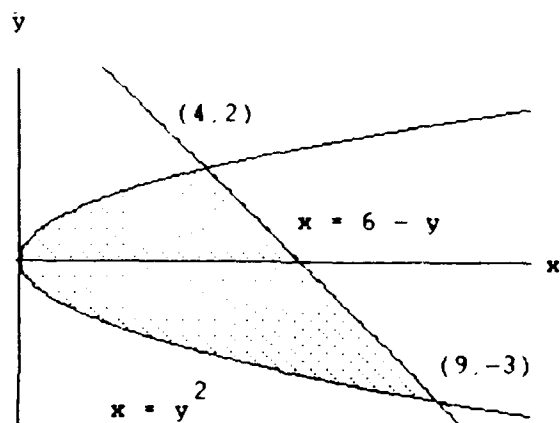
The area of the region is .719.

Example 15

Find the area bounded by the parabola $y^2=x$ and the line $y=6-x$.

Solution.

STEP 1: The graphs. We graph the curves together. The right-hand curve is $x=6-y$, so $f(y)=6-y$. The left-hand curve is $x=y^2$, so $g(y)=y^2$.



STEP 2: The limits of integration. The lower limit of integration is the y-coordinate of the lower point where the parabola crosses the line. The upper limit is the y-coordinate of the upper point where the parabola and the line cross. We find these points by solving the equations $x=y^2$ and $x=6-y$ simultaneously for y:

$$\begin{aligned}
 y^2 &= 6 - y && \text{(Equate } f(y) \text{ and } g(y)) \\
 y^2 + y - 6 &= 0 && \text{(Transpose)} \\
 (y+3)(y-2) &= 0 && \text{(Factor)} \\
 y &= -3, y = 2. && \text{(Solve)}
 \end{aligned}$$

The upper limit of integration is 2 while the lower limit is -3. The limits of integration are $a=-3$ and $b=2$.

STEP 3: The formula for $f(y)-g(y)$.

$$f(y) - g(y) = 6 - y - y^2$$

STEP 4: Integrate.

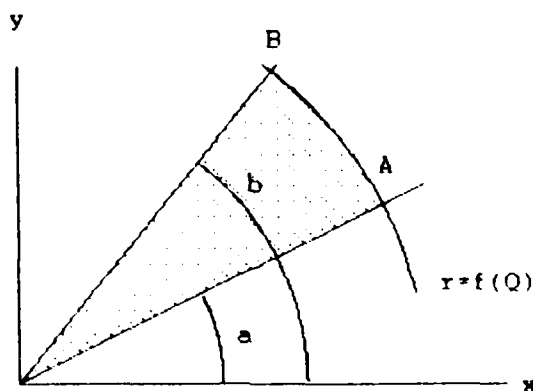
$$\begin{aligned}\text{Area} &= \int_a^b [f(y) - g(y)] dy = \int_{-3}^2 [6 - y - y^2] dy = \left| 6y - \frac{y^2}{2} - \frac{y^3}{3} \right|_{-3}^2 \\ &= \left[6(2) - \frac{(2)^2}{2} - \frac{(2)^3}{3} \right] - \left[6(-3) - \frac{(-3)^2}{2} - \frac{(-3)^3}{3} \right] \\ &= \left[12 - 2 - \frac{8}{3} \right] - \left[-18 - \frac{9}{2} + 9 \right] = 20.833\end{aligned}$$

The area bounded by the curves is 20.833.

III. POLAR COORDINATE SYSTEM

This section shows how to calculate areas of plane regions and lengths of curves. The general methods for setting up the integrals are the same as for cartesian coordinates, although the resulting formulas are somewhat different.

A. Area in the Plane



We calculate areas of plane regions like the shaded one above using polar coordinates.

Area Between the Origin and $r=f(Q)$, Q between a and b :

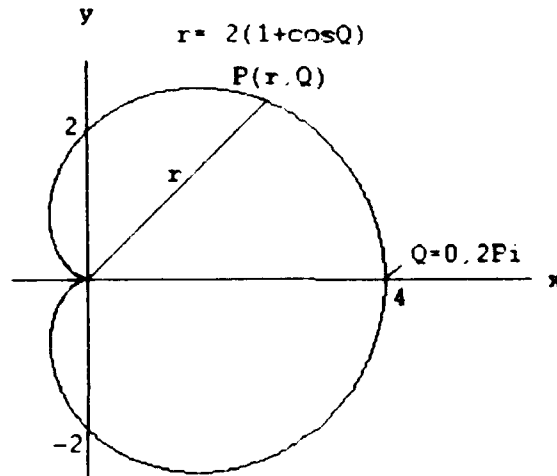
$$\text{Area} = \int_a^b \frac{1}{2} r^2 dQ.$$

Example 16

Find the area of the region enclosed by the cardioid $r=2(1+\cos Q)$.

Solution.

STEP 1: The graph. We graph the cardioid and determine that the radius OP sweeps out the region exactly once as Q runs from 0 to 2π .



STEP 2: The limits of integration. The limits of integration are $a=0$ and $b=2\pi$.

STEP 3: Determine the integrand.

$$\begin{aligned} \frac{1}{2} r^2 &= \frac{1}{2} (4) (1 + \cos Q)^2 = 2(1 + 2\cos Q + \cos^2 Q) \\ &= 2 + 4\cos Q + 2\left[\frac{1 + \cos 2Q}{2}\right] = 3 + 4\cos Q + \cos 2Q \end{aligned}$$

STEP 4: Integrate.

$$\begin{aligned} \text{Area} &= \int_0^{2\pi} [3 + 4\cos Q + \cos 2Q] dQ = 3Q + 4\sin Q + \left[\frac{\sin 2Q}{2}\right] \Big|_0^{2\pi} \\ &= 6\pi + 0 + 0 = 6\pi \end{aligned}$$

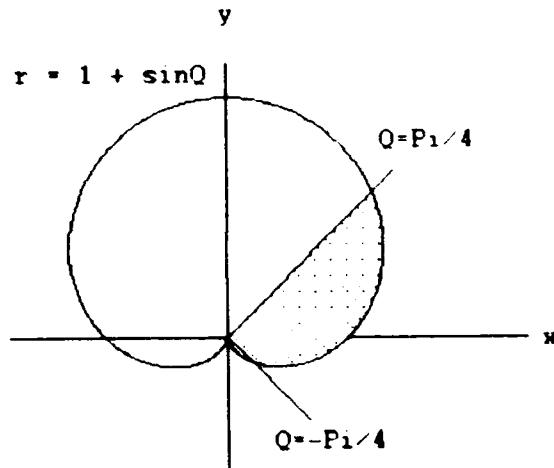
The area of the region is 6π .

Example 17

Find the area of the region bounded by the rays $Q=-\pi/4$ and $Q=\pi/4$, and the graph of the cardioid $r=1+\sin Q$.

Solution.

STEP 1: The graph. The region is the portion of the cardioid swept out by the radius of length $r=1+\sin Q$ as Q increases from $Q=-\pi/4$ to $Q=\pi/4$.



STEP 2: The limits of integration. The limits are given:

$$a = -\pi/4 \text{ and } b = \pi/4.$$

STEP 3: Determine the integrand.

$$\begin{aligned} \frac{1}{2} r^2 &= \frac{1}{2} [1 + \sin Q]^2 = \frac{1}{2} [1 + 2\sin Q + \sin^2 Q] \\ &= \frac{1}{2} [1 + 2\sin Q + (\frac{1}{2} - \frac{1}{2} \cos 2Q)] \\ &= \frac{1}{2} [\frac{3}{2} + 2\sin Q - \frac{1}{2} \cos 2Q] \\ &= \frac{3}{4} + \sin Q - \frac{1}{4} \cos 2Q \end{aligned}$$

STEP 4: Integrate.

$$\begin{aligned} \text{Area} &= \int_a^b \frac{1}{2} r^2 dQ = \int_{-\pi/4}^{\pi/4} [\frac{3}{4} + \sin Q - \frac{1}{4} \cos 2Q] dQ \\ &= \left[\frac{3}{4} Q - \cos Q - \frac{1}{8} \sin 2Q \right]_{-\pi/4}^{\pi/4} = .928 \end{aligned}$$

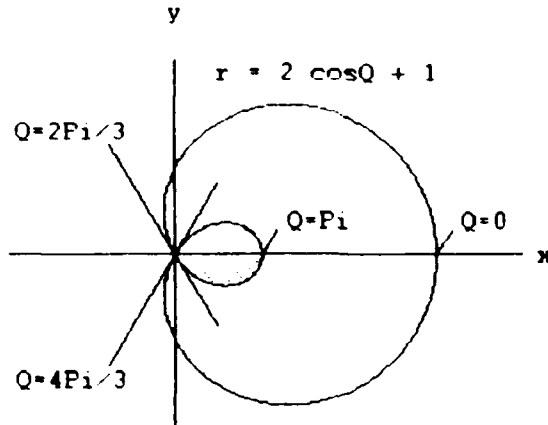
The area of the region is .928.

Example 18

Find the area inside the smaller loop of the limaçon $r = 2\cos Q + 1$.

Solution.

STEP 1: The graph. After sketching the curve we see that the smaller loop is traced out by the point (r, Q) as Q increases from $Q=2\pi/3$ to $Q=4\pi/3$.



STEP 2: The limits of integration. Since the curve is symmetric about the x -axis, we may calculate the area of the shaded half of the inner loop by integrating from $Q=2\pi/3$ to $Q=\pi$. The limits are then $a=2\pi/3$ and $b=\pi$. The area we seek will be twice the value of the resulting integral:

$$\text{Area} = 2 \int_{2\pi/3}^{\pi} \frac{1}{2} r^2 dQ = \int_{2\pi/3}^{\pi} r^2 dQ.$$

STEP 3: Determine the integrand.

$$\begin{aligned} r^2 &= (2\cos Q + 1)^2 &= 4\cos^2 Q + 1 \\ & &= 4\left[\frac{1+\cos 2Q}{2}\right] + 4\cos Q + 1 \\ & &= 2 + 2\cos 2Q + 4\cos Q + 1 \\ & &= 3 + 2\cos 2Q + 4\cos Q \end{aligned}$$

STEP 4: Integrate.

$$\text{Area} = \int_{2\pi/3}^{\pi} [3 + 2\cos 2Q + 4\cos Q] dQ$$

$$\begin{aligned}
&= \left| 3Q + \sin 2Q + 4 \sin Q \right|_{2\pi/3}^{\pi} \\
&= 3\pi - \left[2\pi - \frac{\sqrt{3}}{2} + \frac{4\sqrt{3}}{2} \right] \\
&= \pi - \frac{3\sqrt{3}}{2}
\end{aligned}$$

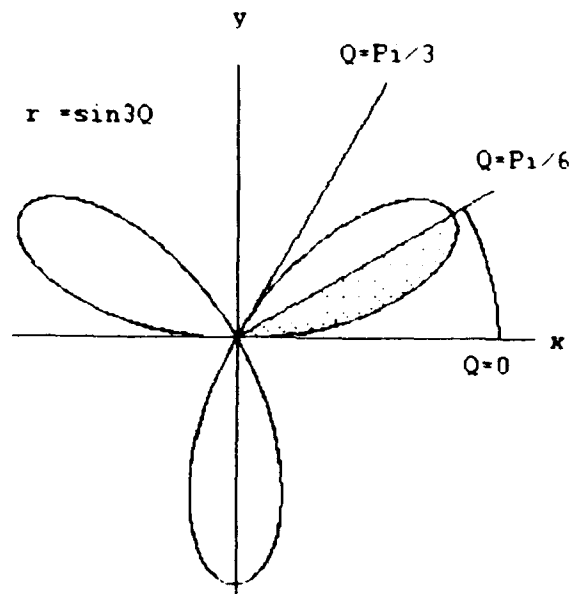
The area of the smaller loop is $\pi - 3\sqrt{3}/2$.

Example 19

Find the area of the region enclosed by the graph of the polar equation $r = \sin 3Q$.

Solution.

STEP 1: The graph. When we graph the region we see that it is a three-leaved rose.



STEP 2: Limits of integration. Since the three leaves are congruent, and since the leaf determined by Q from $Q=0$ to $Q=\pi/3$ is symmetric about the ray $Q=\pi/6$, we may obtain the area of the figure by calculating the area of the half leaf determined by $Q=0$ to $Q=\pi/6$ and multiplying the result by 6. The limits of integration are $a=0$ and $b=\pi/6$.

$$\text{Area} = 6 \int_0^{\pi/6} \frac{1}{2} r^2 dQ$$

STEP 3: Determine the integrand.

$$\begin{aligned} 6 \left[\frac{1}{2} r^2 \right] &= 3r^2 = 3(\sin 3Q)^2 \\ &= 3 \left[\frac{1}{2} - \frac{1}{2} \cos 6Q \right] = \frac{3}{2} - \frac{3}{2} \cos 6Q \end{aligned}$$

STEP 4: Integrate.

$$\text{Area} = \int_0^{\pi/6} \left[\frac{3}{2} - \frac{3}{2} \cos 6Q \right] dQ = \left[\frac{3}{2} Q - \frac{1}{4} \sin 6Q \right] \Big|_0^{\pi/6} = \frac{\pi}{4}$$

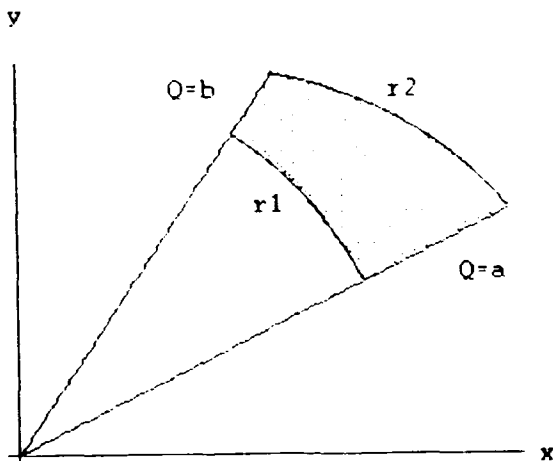
The area of the region is $\pi/4$.

Example 20

Find the area of the region enclosed by the lemniscate $r^2 = \cos 2Q$.

Solution.

STEP 1: The graph. By extracting square roots on both sides of the equation we obtain $r = \sqrt{\cos 2Q}$ and we graph the figure.



STEP 2: The limits of integration. We note that the function r is defined only for Q between $Q=-\pi/4$ and $Q=\pi/4$ and for Q between $Q=3\pi/4$ and $Q=5\pi/4$. As Q ranges through these two intervals the two lobes of the lemniscate are traced out. The rays $Q=-\pi/4$ and $Q=\pi/4$ determine half the area of the lemniscate. We may therefore integrate from $Q=-\pi/4$ to $Q=\pi/4$ to obtain the area of one lobe and double the result. The limits are then $a=-\pi/4$ and $b=\pi/4$.

STEP 3: Determine the integrand.

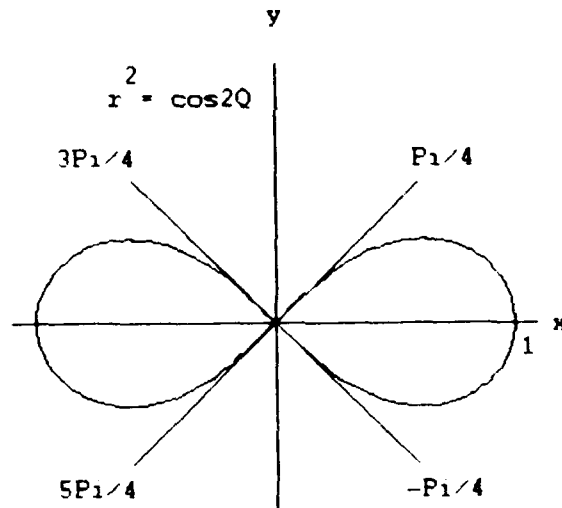
$$r^2 = \cos 2Q$$

STEP 4: Integrate.

$$\text{Area} = 2 \int_{-\pi/4}^{\pi/4} \cos 2Q dQ = \left. \frac{1}{2} \sin 2Q \right|_{-\pi/4}^{\pi/4} = 1$$

The area of the region is 1.

B. Area of Region which lies Between two Polar Curves



To find the area of a region which lies between two polar curves from $Q=a$ to $Q=b$, we subtract the integral of $(1/2)r_1^2 dQ$ from the integral of $(1/2)r_2^2 dQ$. This leads to the following formula.

Area of the Region: $r_1(Q)$ less than or equal to r and r less than or equal to $r_2(Q)$, Q from $Q=a$ to $Q=b$

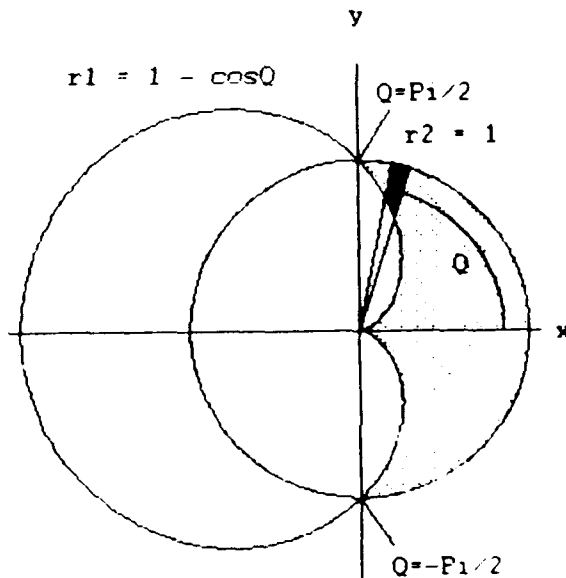
$$\text{Area} = \int_a^b \frac{1}{2} r_2^2 dQ - \int_a^b \frac{1}{2} r_1^2 dQ = \int_a^b \frac{1}{2} (r_2^2 - r_1^2) dQ.$$

Example 21

Find the area of the region that lies inside the circle $r=1$ and outside the cardioid $r=1-\cos Q$.

Solution.

STEP 1: The graphs. We sketch the region to determine its boundaries and find the limits of integration. The outer curve is $r_2=1$, and the inner curve is $r_1=1-\cos Q$.



STEP 2: Limits of integration. We see that Q runs from $-\pi/2$ to $\pi/2$. Because of symmetry we find the area for $Q=0$ to $Q=\pi/2$. The limits of integration are $a=-\pi/2$ and $b=\pi/2$.

$$\begin{aligned} \text{Area} &= \int_{-\pi/2}^{\pi/2} \frac{1}{2} (r_2^2 - r_1^2) dQ = 2 \int_0^{\pi/2} \frac{1}{2} (r_2^2 - r_1^2) dQ \\ &= \int_0^{\pi/2} (r_2^2 - r_1^2) dQ \end{aligned}$$

STEP 3: Determine the integrand.

$$r_2^2 - r_1^2 = 1 - (1 - 2\cos Q + \cos^2 Q) = 2\cos Q - \cos^2 Q$$

$$= 2\cos Q - \frac{\cos 2Q + 1}{2}$$

STEP 4: Integrate.

$$\begin{aligned} \text{Area} &= \int_0^{\pi/2} \left[2\cos Q - \frac{\cos 2Q + 1}{2} \right] dQ \\ &= \left[2\sin Q - \frac{\sin 2Q}{4} - \frac{Q}{2} \right]_0^{\pi/2} = 2 - \frac{\pi}{4} \end{aligned}$$

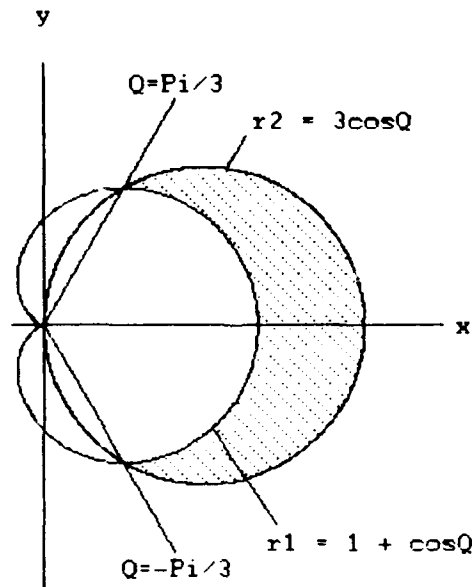
The area of the region is $2 - \pi/4$.

Example 22

Find the area of the region lying inside the circle $r=3\cos Q$ and outside the cardioid $r=1+\cos Q$.

Solution.

STEP 1: The graphs. We sketch the graphs of the two equations $r^2=3\cos Q$ and $r_1=1+\cos Q$.



STEP 2: The limits of integration. By solving the two equations simultaneously, we find the point of intersection:

$$3\cos Q = 1 + \cos Q \quad (\text{Equate } r_1 \text{ and } r_2)$$

$$2\cos Q = 1 \quad (\text{Transpose})$$

$$\cos Q = \frac{1}{2} \quad (\text{Divide})$$

$$Q = \pi/3, Q = -\pi/3. \quad (\text{Solve})$$

The limits of integration are $a = -\pi/3$ and $b = \pi/3$.

STEP 3: Determine the integrand.

$$\begin{aligned} \frac{1}{2}(r_2^2 - r_1^2) &= \frac{1}{2}[(3\cos Q)^2 - (1 + \cos Q)^2] \\ &= \frac{1}{2}[9\cos^2 Q - (1 + 2\cos Q + \cos^2 Q)] \\ &= \frac{1}{2}[8\cos^2 Q - 1 - 2\cos Q] \\ &= 4\cos^2 Q - \frac{1}{2} - \cos Q \\ &= 4\left[\frac{1}{2} + \frac{1}{2}\cos 2Q\right] - \frac{1}{2} - \cos Q \\ &= 2 + 2\cos 2Q - \frac{1}{2} - \cos Q \\ &= \frac{3}{2} + 2\cos 2Q - \cos Q \end{aligned}$$

STEP 4: Integrate.

$$\begin{aligned} \text{Area} &= \int_{-\pi/3}^{\pi/3} \left[\frac{3}{2} + 2\cos 2Q - \cos Q\right] dQ \\ &= \left[\frac{3}{2}Q + \sin 2Q - \sin Q\right]_{-\pi/3}^{\pi/3} = \text{Pi} \end{aligned}$$

The area of the region is π .

C. The Length of a Curve

We calculate the length of a curve $r=f(Q)$, Q from a to b by expressing the differential $ds=\sqrt{(dx^2+dy^2)}$ in terms of Q and integrating from a to b .

If $r=f(Q)$ has a continuous first derivative for Q from a to b and if the point $P(r,Q)$ traces the curve $r=f(Q)$ exactly once as Q runs from a to b , then the length of the curve is given by:

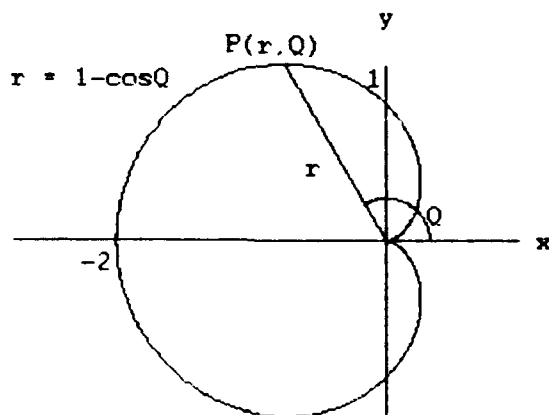
$$\text{Length} = \int_a^b \sqrt{r^2 + \left(\frac{dr}{dQ}\right)^2} dQ.$$

Example 23

Find the length of the cardioid $r=1-\cos Q$.

Solution.

STEP 1: The graph. We sketch the cardioid to determine the limits of integration. The point $P(r,Q)$ starts at the origin and traces the curve once, counterclockwise.



STEP 2: Limits of integration. We see that Q runs from 0 to 2π , so the limits of integration are $a=0$ and $b=2\pi$.

STEP 3: Determine the integrand.

$$r=1-\cos Q, \quad \frac{dr}{dQ} = \sin Q,$$

$$r^2 + \left(\frac{dr^2}{dQ}\right) = (1 - \cos Q)^2 + (\sin Q)^2 = 1 - 2\cos Q + \cos^2 Q + \sin^2 Q$$

$$= 1 - 2\cos Q + 1 = 2 - 2\cos Q$$

STEP 4: Integrate.

$$\text{Length} = \int_a^b \sqrt{r^2 + \left(\frac{dr^2}{dQ}\right)} dQ = \int_0^{2\pi} \sqrt{2 - 2\cos Q} dQ$$

$$= \int_0^{2\pi} \sqrt{2(1 - \cos Q)} dQ = \int_0^{2\pi} \sqrt{2\left(2\sin^2 \frac{Q}{2}\right)} dQ$$

$$= \int_0^{2\pi} \sqrt{4\sin^2 \frac{Q}{2}} dQ = \int_0^{2\pi} 2 \left[\sin \frac{Q}{2} \right] dQ = \int_0^{2\pi} 2\sin \frac{Q}{2} dQ$$

$$= \left[-4\cos \frac{Q}{2} \right]_0^{2\pi} = [4 + 4] = 8$$

The length of the curve is 8.

APPENDIX B

TUTORIAL

A. INTRODUCTION

This tutorial instructs the user on how to start **Mathematica** and how to load and use the program. The tutorial includes three example problems that are intended to show the user how the program works. The first example problem is an integral problem given in cartesian coordinates. In the first problem, the program is used to find the area of a region by first integrating with respect to the variable x and then with respect to the variable y . The second problem is an integral problem given in polar coordinates. In this problem the program finds the area between a polar curve and the origin. The final example problem is also given in polar coordinates. In this problem the program is used to find the area of the region bounded by two polar curves. These three example problems are representative problems that the program can solve. They show some of the program's many capabilities. The tutorial, while not comprehensive, gives the user some idea of the types of problems the program can solve. It demonstrates how the program prompts the user for input and how that input is entered.

B. STARTING MATHEMATICA AND LOADING PROGRAM

To start **Mathematica**, double-click the **Mathematica** icon. **Mathematica** documents are called Notebooks. When you start **Mathematica**, an empty Notebook window appears on your screen. As soon as you start typing, a cell is created, indicated by a cell bracket along the right edge of the Notebook window. Each piece of information in a Notebook is contained in a separate cell. To load the program and start the kernel first type: `<<integral.ma.`

You now must activate the cell. There are several ways to do this. You can use the Mouse and select **Evaluate Selection**, in the **Action** menu which appears at the top of the screen (see Figure 18 on page 96). The three keyboard equivalents are:

1. Shift+Enter
2. Insert
3. 5 on the numeric keypad.

Once you have activated the cell you are ready to use the program.

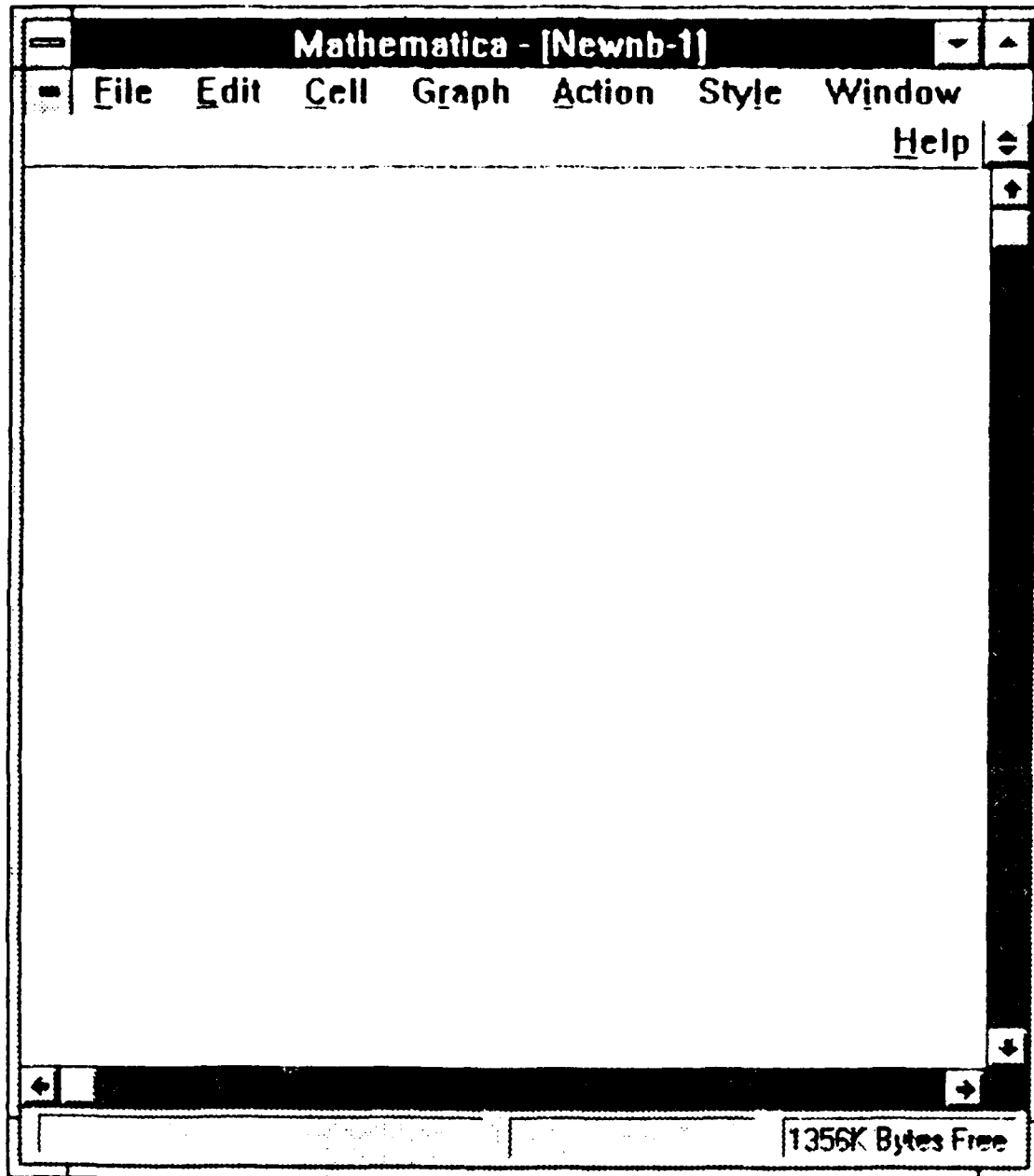


Figure 18

C. EXAMPLE 1

Find the area of the region bounded by the functions $y = 5x$, $y = x$ and $y = 4 - x^2$ in the first quadrant.

* Press the down arrow to create a new cell and then type **graph**.

* For this problem press the Insert key to activate the cell.

* You will see appear the message:

To display a graph that is in the lab book type the example number, otherwise type n.

Because you are going to enter our own example type n and press Enter.

* You will see the message:

Type 1 for cartesian coordinate system, or type 2 for polar coordinate system.

Since your first example is a problem given in cartesian coordinates type 1 and press Enter.

* You will see the message:

To just plot curves type y, otherwise type n.

Since you do not want to just plot the functions type n.

* You will see the message:

Enter the functions which define the region's boundaries. When finished type n.

The functions must be functions of the independent variable x and are entered one at a time. They can be inputted in any order. You are now ready to enter the functions. Type $5x$ then press Enter, type x then press Enter, type $4 - x^2$ then press Enter.

* You will see the message:

If any of the limits are known type y, otherwise type n.

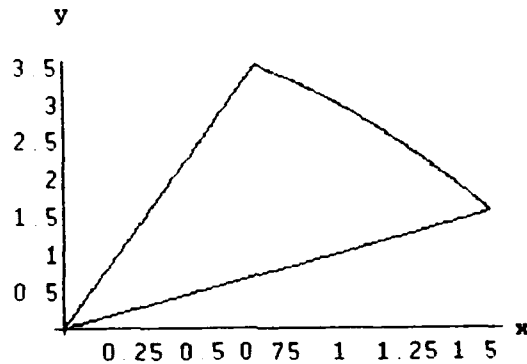
Since none of the limits are explicitly given you type n.

* You will see the message:

If the region is restricted to the right half plane type y, otherwise type n.

Because we are told in the problem statement that the region is restricted to the first quadrant type y.

- * The region is then plotted and the program solves the problem. You will see the following output:



The limits of integration for the first region are $x = 0$ and $x = 0.702$

The upper curve for the first region is $y = 5x$

The lower curve for the first region is $y = x$

The area of the first region is 0.985608

The limits of integration for the second region are $x = 0.702$ and $x = 1.562$

The upper curve for the second region is $y = 4 - x^2$

The lower curve for the second region is $y = x$

The area of the second region is 1.31145

The area of the total region is 2.29706

- * All of the functions are invertible and and it is appropriate to integrate the problem with respect to the variable y. As a result, after a brief delay you will see the message:

Type y if you wish to integrate the same problem with respect to y, otherwise type n.

Type **y** and you will see the following output:

The limits of integration for the first region are $y = 0$
and $y = 1.56$

The right curve for the first region is $x = y$

The left curve for the first region is $x = \frac{y}{5}$

The area of the first region is 0.973443

The limits of integration for the second region are $y = 1.56$
and $y = 3.51$

The right curve for the second region is $x = \text{Sqrt}[4-y]$

The left curve for the second region is $x = \frac{y}{5}$

The area of the second region is 1.32362

The area of the total region is 2.29706

D. EXAMPLE 2

Find the area of one leaf of the three-leaf rose formed by the polar curve $r = \sqrt{\sin 3\theta}$.

- * Press the down arrow to create a new cell and then type **graph**.
- * Activate the cell by using the Mouse to select **Evaluate Selection** from the **Action** menu.
- * You will see appear the message:

To display a graph that is in the lab book type the example number, otherwise type n.

Because you are going to enter our own example type **n** and press **Enter**.

- * You will see the message:
Type 1 for cartesian coordinrte system, or type 2 for polar coordinate system.

Since your second example is a problem given in polar coordinants type **2** and press **Enter**.

- * You will see the message:
To just plot curves type y, otherwise type n.
Since you do not want to just plot the polar curve type **n**.

* You will see the message:

Type 1 for area between the origin and a polar curve, or
type 2 for area between two polar curves.

Type 1 since your problem is to find the area between the
origin and the polar curve $r = \sqrt{\sin 3\theta}$ for one leaf.

* You will see the message:

Enter the polar curve which bounds the region.

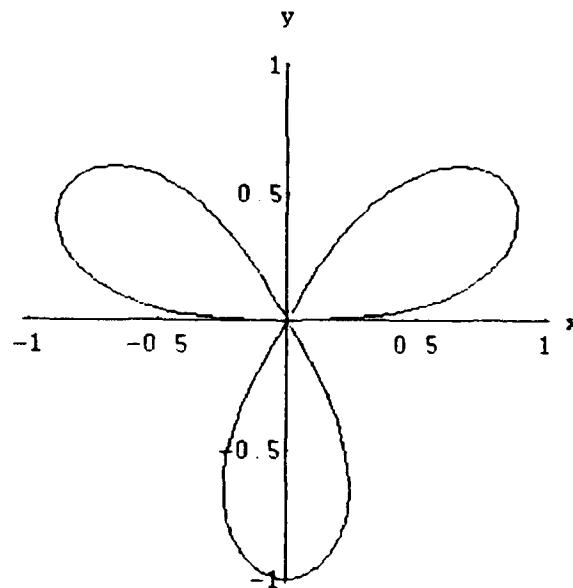
Type `sqrt[sin[3Q]]`.

* You will see the message:

If the limits are known type y, otherwise type n.

Since the limits are not known type n.

* The polar curve is then plotted. You will see the
following output:



The polar curve intersects the origin at Q:

$$\left(0, \frac{\pi}{3}, \frac{2\pi}{3}, \pi, \frac{4\pi}{3}, \frac{5\pi}{3}, 2\pi\right)$$

* After a brief delay you will see the message:

The polar curve is plotted from $Q=0$ to $Q=2\text{ Pi}$. If you want to change the limits type y , otherwise type n .

Since you are only interested in the area of one leaf type y .

* You will see the message:

Input new lower limit.

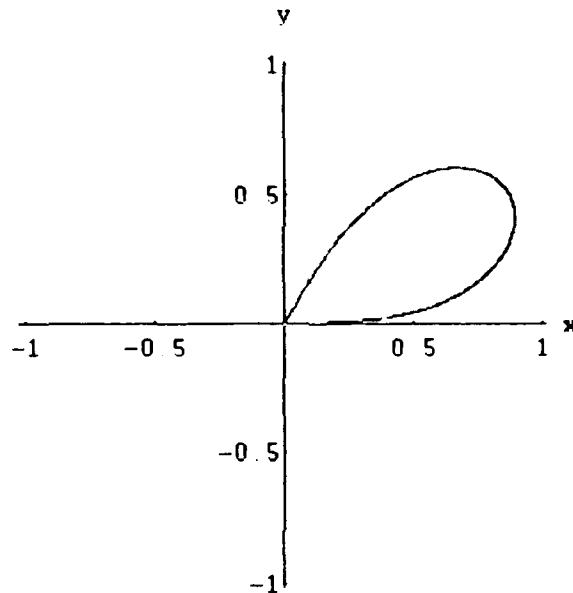
Type 0.

* You will see the message:

Input new upper limit.

Type $\text{pi}/3$.

* The region in which you are interested is then plotted. You will see the following output:



The limits of integration for the region which is bounded by the polar curve $r = \text{Sqrt}[\sin[3Q]]$ and the origin are:

$$a=0 \text{ and } b=\frac{\text{Pi}}{3}$$

The area bounded by the polar curve and the origin is .333

E. EXAMPLE 3

Find the area of the region bounded by the circle $r = 1$ and the cardioid $r = 1 - \cos \theta$ if the outer curve is the cardioid and the inner curve is the circle.

* Press the down arrow to create a new cell and then type graph.

* Activate the cell by pressing Shift+Enter.

* You will see appear the message:

To display a graph that is in the lab book type the example number, otherwise type n.

Because you are going to enter our own example type n and press Enter.

* You will see the message:

Type 1 for cartesian coordinate system, or type 2 for polar coordinate system.

Since your third example is a problem given in polar coordinates type 2 and press Enter.

* You will see the message:

To just plot curves type y, otherwise type n.

Since you do not want to just plot the polar curves type n.

* You will see the message:

Type 1 for area between the origin and a polar curve, or type 2 for area between two polar curves.

Type 2 since your problem is to find the area between two polar curves.

* You will see the message:

Enter one of the polar curves which bounds the region.

Type $1 - \cos[\theta]$.

* You will see the message:

Enter the other polar curve which bounds the region.

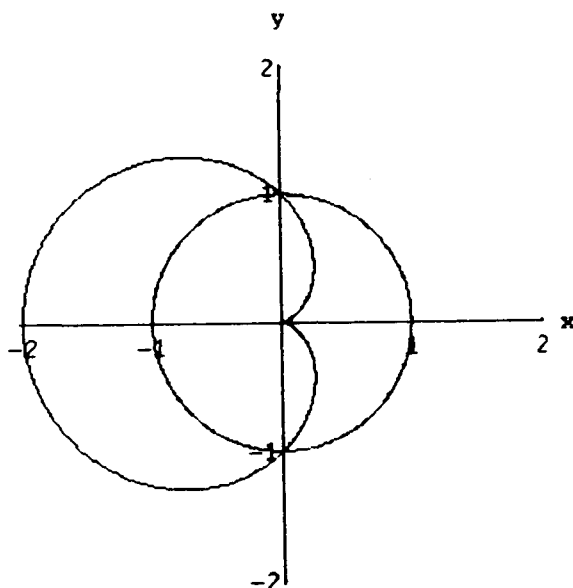
Type 1.

* You will see the message:

If the limits are known type y, otherwise type n.

Since we do not know where the two curves intersect and what the limits of integration are type n.

* The two polar curves will then be plotted. You will see the following output:



The two polar curves intersect at Q:

$$\left(\frac{\pi}{2}, \frac{3\pi}{2}\right)$$

* After a breif delay you will see the message:

Input lower limit.

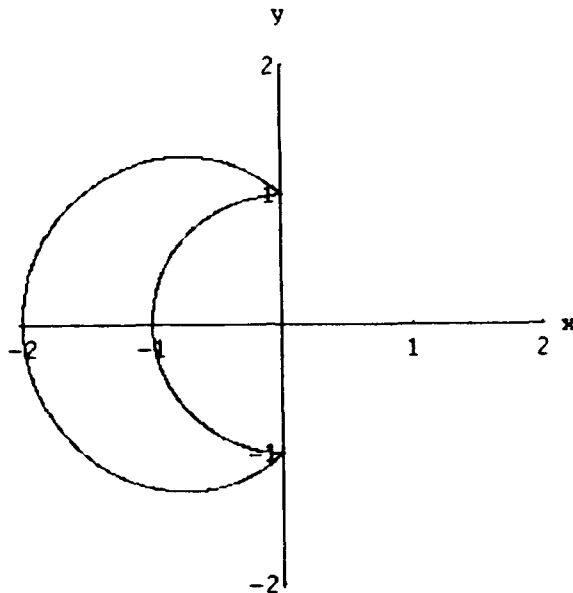
Since the two curves intersect and bound two different regions you must be careful in selecting the correct limits. Type $\pi/2$.

* You will see the message:

Input upper limit.

Type $3/2 \text{ pi}$.

* The region is then plotted. You will see the following output:



The outer polar curve is $r = 1 - \cos[\theta]$

The inner polar curve is $r = 1$

The limits of integration are $a = \frac{\pi}{2}$ and $b = \frac{3\pi}{2}$

The area of the region bounded by the two polar curves is 2.7854

APPENDIX C - PROGRAM CODE

I. FILE CART.MA

(* Copyright Dennis A. Polaski, May 4, 1993 *)

```
cart:=
Module[(),
cltranscend; ln[x_]=.;restrictionR=n; restrictionL=n;
endpoints={}; intersect={};
myfunc={}; rightlimit=n;leftlimit=n; type2D=1;

While[type2D==1, {
c2func=Input["Enter the functions which
define the region's boundaries. When finished type n.

y ="];
If[c2func !=n, {
functotal=ToExpression[StringJoin["y==","c2func"]];
AppendTo[myfunc,functotal]}, type2D=0];
If[Length[myfunc]==3,type2D=0]
}];

myfunc=myfunc /. sqrt->Sqrt;
myfunc=myfunc /. pi->Pi;

If[Length[myfunc]==2, {
limitsknown=Input["If any of the limits is known type
y, otherwise type n."];

While[limitsknown ==y,
leftlimit=Input["Input left limit if known, otherwise
type n.

x ="];
leftlimit=leftlimit /. sqrt->Sqrt;
leftlimit=leftlimit /. pi->Pi;
If[leftlimit != n, AppendTo[myfunc,
ToExpression[StringJoin["x==","leftlimit"]]]];
rightlimit=Input["Input right limit if known, otherwise
type n.
```



```

x ="];
rightlimit=rightlimit /. sqrt->Sqrt;
rightlimit=rightlimit /. pi->Pi;
If[rightlimit!= n, AppendTo[myfunc,
ToExpression[StringJoin["x==", "rightlimit"]]]];
limitsknown = n] ;
]];

If[leftlimit === n && (rightlimit >0 || rightlimit===n),
restrictionR=Input["If the region is restricted to the
right half plane type y, otherwise type n."]
];

If[rightlimit === n && restrictionR===n &&
(leftlimit <0 || leftlimit===n),
restrictionL=Input["If the region is restricted to the
left half plane type y, otherwise type n."]
];

last=Length[myfunc];
For[ii=1, ii<= last,
  {ival=ii; ++ii;

For[jj=1, jj<=last,
  {jval=jj; ++jj;

If[jval!=ival &&
(myfunc[[ival,1]]===y || myfunc[[jval,1]]===y),
{
ln[x_]:=Log[x];
If[myfunc[[ival,1]]===y && myfunc[[jval,1]]===y,
{
If[{myfunc[[ival]], myfunc[[jval]]}!=
({myfunc[[ival]], myfunc[[jval]] /. Log[x_]->x}),
{theroots={nSolve[{myfunc[[ival]],myfunc[[jval]]}]}}
},
{
rtranscend;
theroots=NSolve[{myfunc[[ival]], myfunc[[jval]]},{x,y}];
theroots=N[{x,y} /. theroots];
theroots=DeleteCases[DeleteCases[DeleteCases[
theroots, {x_Complex,y_}],{x_,y_Complex}],
{x_Complex,y_Complex}];

```

```

If[Length[theroots]==0,{
theroots=Solve[{myfunc[[ival]], myfunc[[jval]]},{x,y}];
theroots=N[{x,y} /. theroots];
theroots=DeleteCases[DeleteCases[DeleteCases[
theroots, {x_Complex,y_}],{x_,y_Complex}],
{x_Complex,y_Complex}]];

cltranscend;

If[theroots!={x,y} && theroots !={} &&
{myfunc[[ival]],myfunc[[jval]]}===
({myfunc[[ival]],myfunc[[jval]]) /. Sqrt[x_]->x),
{
rmtranscend;
theroots=Table[FindRoot[myfunc[[ival,2]]-
myfunc[[jval,2]]==0, {x,theroots[[m,1]}],
{m,Length[theroots]}];
theroots=x /. theroots;
theroots=Table[{theroots[[m]],(myfunc[[ival,2]] /.
x->theroots[[m]])},{m,Length[theroots]}];
temproots=
Table[{TrueQ[(Abs[myfunc[[ival,2]]-myfunc[[jval,2]])] /.
x -> theroots[[s,1]] <=.5]},{s,Length[theroots]}];
theroots=Transpose[{theroots,temproots}];
theroots=Transpose[DeleteCases[theroots,{_,(False)},1]][[1]];
If[theroots== {}[[1]],theroots={}];

x=.
}];
}];
}];
rmtranscend;

If[myfunc[[ival,1]]===y && myfunc[[jval,1]]===x,
theroots=N[{{myfunc[[jval,2]],(myfunc[[ival,2]]/.
x->myfunc[[jval,2]])}}];

If[myfunc[[ival,1]]===x && myfunc[[jval,1]]===y,
theroots=N[{{myfunc[[ival,2]],(myfunc[[jval,2]]/.
x->myfunc[[ival,2]])}}];

theroots=DeleteCases[DeleteCases[DeleteCases[
theroots, {x_Complex,y_}],{x_,y_Complex}],
{x_Complex,y_Complex}];

theroots=ReplaceAll[theroots, 0. -> 0];
If[restrictionR===y,
theroots=Select[theroots, (#[[1]]>=0)&]];
If[restrictionL===y,
theroots=Select[theroots, (#[[1]]<=0)&]];

```

```

If[theroots != {x,y} && theroots != {},{
endpoints=Join[endpoints, theroots] }];

]];
If[leftlimit !=n, endpoints=Select[endpoints, (#[[1]]>=
leftlimit)&]];
If[rightlimit !=n, endpoints=Select[endpoints, (#[[1]]<=
rightlimit)&]];

]];
endpoints=N[Round[endpoints*1000]/1000];
endpoints=Union[endpoints, {}];
intersect=Join[intersect, endpoints];
intersect=Union[intersect, {}];

ycoord=Sort[Union[Transpose[Union[intersect, {}]][[2]], {}]];
ycoord=N[Union[Round[ycoord*100]]/100];
mm=Transpose[{endpoints[[1]], endpoints[[2]]}];
mnew=Sort[mm[[1]]; x=.;

If[ival==1, {
If[mnew[[1]] != mnew[[2]],
{line1=Plot[myfunc[[ival,2]],{x,mnew[[1]],mnew[[2]]},
DisplayFunction-> Identity]; line11=line1[[1,1,1,1]]},
{line1=Show[Graphics[{Line[endpoints]}]},
DisplayFunction-> Identity]; line11=endpoints}
}];

If[ival==2, {
If[mnew[[1]] != mnew[[2]],
{line2=Plot[myfunc[[ival,2]],{x,mnew[[1]], mnew[[2]]},
DisplayFunction-> Identity]; line22=line2[[1,1,1,1]]},
{line2=Show[Graphics[{Line[endpoints]}]},
DisplayFunction-> Identity]; line22=endpoints}
}];

If[ival==3, {
If[mnew[[1]] != mnew[[2]],
{line3=Plot[myfunc[[ival,2]],{x,mnew[[1]], mnew[[2]]},
DisplayFunction-> Identity]; line33=line3[[1,1,1,1]]},
{line3=Show[Graphics[{Line[endpoints]}]},
DisplayFunction-> Identity]; line33=endpoints}
}];

If[ival==4, {
If[mnew[[1]] != mnew[[2]],
{line4=Plot[myfunc[[ival,2]],{x,mnew[[1]], mnew[[2]]},
DisplayFunction-> Identity]; line44=line4[[1,1,1,1]]},
{line4=Show[Graphics[{Line[endpoints]}]},
DisplayFunction-> Identity]; line44=endpoints}
}];

```

```

    });
If[ival==5, (
If[mnew[[1]] != mnew[[2]],
{line5=Plot[myfunc[[ival,2]],{x,mnew[[1]], mnew[[2]]},
DisplayFunction-> Identity]; line55=line[[1,1,1,1]],
{line5=Show[Graphics[{Line[endpoints]}]},
DisplayFunction-> Identity]; line55=endpoints}
)];
If[ival==6, (
If[mnew[[1]] != mnew[[2]],
{line5=Plot[myfunc[[ival,2]],{x,mnew[[1]], mnew[[2]]},
DisplayFunction-> Identity]; line66=line6[[1,1,1,1]],
{line6=Show[Graphics[{Line[endpoints]}]},
DisplayFunction-> Identity]; line66=endpoints}
)];
endpoints={}

)
];
If[ival==2, {Show[line1,line2,DisplayFunction->
$DisplayFunction];
thelines={line11,line22}}];
If[ival==3, {Show[line1,line2,line3,DisplayFunction->
$DisplayFunction];
thelines={line11,line22,line33}}];
If[ival==4, {Show[line1,line2,line3,line4,DisplayFunction->
$DisplayFunction];
thelines={line11,line22,line33,line44}}];
If[ival==5, {Show[line1,line2,line3,line4,line5,
DisplayFunction-> $DisplayFunction];
thelines={line11,line22,line33,line44,line55}}];
If[ival==6, {Show[line1,line2,line3,line4,line5,line6,
DisplayFunction-> $DisplayFunction];
thelines={line11,line22,line33,line44,line55,line66}}];

intersect=Transpose[intersect];
intersect=Sort[intersect[[1]]];
intersect=ReplaceAll[intersect,0->0.];

xcoord=Union[intersect,{}];
xcoord=Union[xcoord/1.];
rmtranscend;

For[kk=1, kk<=Length[xcoord]-1,
{kval=kk; ++kk;
xx=(xcoord[[kval]]+xcoord[[kval+1]])/2;
integrfunc=Select[thelines,(xx>#[[1,1]] && xx<#[[-1,1]])&];
place1=Position[thelines, integrfunc[[1]][[1]][[1]]];
place2=Position[thelines, integrfunc[[2]][[1]][[1]]];

```

```

x=. ; rctranscend;
area=NIntegrate[Evaluate[myfunc[[place1,2]]-myfunc[[place2,2
]],
{x, xcoord[[kval]], xcoord[[kval+1]]}]];
ctranscend; ln[x_]=.;
If[area < 0,
{upper=myfunc[[place2]]; lower=myfunc[[place1]]},
{upper=myfunc[[place1]]; lower=myfunc[[place2]]}]];

If[kval==1 && Length[xcoord]==2,
{Print["The limits of integration are x = ",
xcoord[[kval]]," and x = ", xcoord[[kval+1]]];
Print["The upper curve is y = ", upper[[2]]];
Print["The lower curve is y = ", lower[[2]]];
Print["The area of the region is ", Abs[area]] }]];

If[kval==1 && Length[xcoord]==3,
{Print["The limits of integration for the first region
are x = ", xcoord[[kval]]," and x = ", xcoord[[kval+1]]];
Print["The upper curve for the first region is y = ",
upper[[2]]];
Print["The lower curve for the first region is y = ",
lower[[2]]];
Print["The area of the first region is ", Abs[area]];
areal=Abs[area]]};

If[kval==2 && Length[xcoord]==3,
{Print["The limits of integration for the second region
are x = ", xcoord[[kval]]," and x = ", xcoord[[kval+1]]];
Print["The upper curve for the second region is y = ",
upper[[2]]];
Print["The lower curve for the second region is y = ",
lower[[2]]];
Print["The area of the second region is ", Abs[area]];
Print["The area of the total region is ", areal+
Abs[area]]}];
]];
inverse;
];

ctranscend:=Module[{},sin[x_]=.; cos[x_]=.; tan[x_]=.;
cot[x_]=.; csc[x_]=.; sec[x_]=.; exp[x_]=.]

transcend:=Module[{},
sin[x_] := x - x^3/6 + x^5/120 - x^7/5040 + x^9/362880 -
x^11/39916800;
cos[x_] := 1 - x^2/2 + x^4/24 - x^6/720 + x^8/40320 -
x^10/3628800 + x^12/479001600 -
x^14/87178291200 + x^16/20922789888000;
tan[x_] := x + x^3/3 + 2x^5/15 + 17x^7/315 + 62x^9/2835 +
1382x^11/155925;
cot[x_] := 1/x -x/3 -x^3/45 - 2x^5/945 - x^7/4725 -

```

```

2x^9/93555;
csc[x_]:= 1/x + x/6 + 7x^3/360 + 31x^5/15120 +
127x^7/604800 + 73x^9/3421440;
sec[x_]:= 1 + x^2/2 + 5x^4/24 + 61x^6/720 + 277x^8/8064 +
50521x^10/3628800;
exp[x_]:= 1 + x + x^2/2 + x^3/6 + x^4/24 + x^5/120 + x^6/720
+
x^7/5040 + x^8/40320 + x^9/362880]

```

```

rmtranscend:=Module[(), sin[x_]:=Sin[x]; cos[x_]:=Cos[x];
tan[x_]:=Tan[x]; cot[x_]:=Cot[x]; csc[x_]:=Csc[x];
sec[x_]:=Sec[x]; exp[x_]:=Exp[x] ]

```

```

nSolve[{x==a_,y==d_+c_*Log[b_*x]}]:=
{a,N[d+c*Log[b*a]]}
nSolve[{y==d_+c_*Log[b_*x],x==a_}]:=
{a,N[d+c*Log[b*a]]}

```

```

nSolve[{y==d1_+c1_*Log[b1_*x],
y==d2_+c2_*Log[b2_*x]}]:=
N[(((b1^c1)/(b2^c2)Exp[d1-d2])^(1/(c2-c1)),d1+c1*Log[b1*x]/.
x->((b1^c1)/(b2^c2)Exp[d1-d2])^(1/(c2-c1)))]

```

```

nSolve[F_List]:={x,y} /.NSolve[F,{x,y}]

```

II. FILE CARTPLOT.MA

(* Copyright Dennis A. Polaski, May 4, 1993 *)

```
cartplot:=Module[{},allyfunc={};allxfunc={};xlines={};
myfunc={};endpoints={};intersect={};
cltranscend;
f[{x___}]:=x;
type2D=1;
```

```
While[type2D==1,
yfunc=Input["Enter the curves, one at a time. When
finished type n.
```

```
                y = "];
yfunc=yfunc /. sqrt->Sqrt;
yfunc=yfunc /. pi->Pi;
```

```
If[yfunc!=n, {
AppendTo[allyfunc, yfunc],
AppendTo[myfunc, ToExpression[StringJoin["y==", "yfunc"]]}],
{type2D=0}]
];
```

```
While[type2D==0,
xfunc=Input["Enter lines perpendicular to the x-axis.
When finished type n.
```

```
                x = "];
xfunc=xfunc /. sqrt->Sqrt;
xfunc=xfunc /. pi->Pi;
```

```
If[xfunc!=n, {
AppendTo[allxfunc, xfunc],
AppendTo[myfunc, ToExpression[StringJoin["x==", "xfunc"]]}],
{type2D=-1}]
];
```

```
xlower=Input["Enter lower limit for x-range of plot."];
xlower=xlower /. sqrt->Sqrt;
xlower=xlower /. pi->Pi;
xupper=Input["Enter upper limit for x-range of plot."];
xupper=xupper /. sqrt->Sqrt;
xupper=xupper /. pi->Pi;
ylower=Input["Enter lower limit for y-range of plot."];
ylower=ylower /. sqrt->Sqrt;
ylower=ylower /. pi->Pi;
yupper=Input["Enter upper limit for y-range of plot."];
yupper=yupper /. sqrt->Sqrt;
yupper=yupper /. pi->Pi;
```

```

If[Length[allxfunc]>=1,{
For[i=1, i<=Length[allxfunc], {ival=i; ++i;
line=Show[Graphics[{
Line[{{allxfunc[[ival]], ylower},{allxfunc[[ival]],
yupper}}] ]], AxesLabel->{"x","y"},DisplayFunction
->Identity];
AppendTo[xlines,line] ]];
}];

rmtranscend;
theplot=Plot[Evaluate[allyfunc],{x, xlower, xupper},
AxesLabel->{"x","y"}, PlotRange->{{xlower,xupper},
{ylower,yupper}},DisplayFunction->Identity];
Show[xlines, theplot,DisplayFunction->$DisplayFunction];

x=.;y=.;
last=Length[myfunc];
For[ii=1, ii<= last,
{ival=ii; ++ii;

For[jj=1, jj<=last,
{jval=jj; ++jj;

cltranscend;
If[{jval!=ival &&
(myfunc[[ival,1]]===y || myfunc[[jval,1]]===y),
{
ln[x_]:=Log[x];
If[myfunc[[ival,1]]===y && myfunc[[jval,1]]===y,
{
If[{myfunc[[ival]], myfunc[[jval]]}!=
({myfunc[[ival]], myfunc[[jval]] /. Log[x_]->x)},
{theroots={NSolve[{myfunc[[ival]],myfunc[[jval]]}]}
},
{
transcend;
theroots=NSolve[{myfunc[[ival]], myfunc[[jval]]},{x,y}];
theroots=N[{x,y} /. theroots];
theroots=DeleteCases[DeleteCases[DeleteCases[
theroots, {x_Complex,y_}],{x_,y_Complex}],
{x_Complex,y_Complex}];

If[Length[theroots]==0,{
theroots=Solve[{myfunc[[ival]], myfunc[[jval]]},{x,y}];
theroots=N[{x,y} /. theroots];
theroots=DeleteCases[DeleteCases[DeleteCases[
theroots, {x_Complex,y_}],{x_,y_Complex}],
{x_Complex,y_Complex}
]
}];
}];
}];
}];

```



```

cltranscend;

If[theroots!=({x,y} && theroots !={} &&
(myfunc[[ival]],myfunc[[jval]])===
((myfunc[[ival]],myfunc[[jval]])) /. Sqrt[x_]->x),
{
rmtranscend;
theroots=Table[FindRoot[myfunc[[ival,2]]-
myfunc[[jval,2]]==0, {x,theroots[[m,1]}],
{m,Length[theroots]}];
theroots=x /. theroots;
theroots=Table[{theroots[[m]],(myfunc[[ival,2]] /.
x->theroots[[m]])}, {m,Length[theroots]}];
temproots=
Table[{TrueQ[(Abs[myfunc[[ival,2]]-myfunc[[jval,2]]) /.
x -> theroots[[s,1]]) <=.5]}, {s,Length[theroots]}];
theroots=Transpose[{theroots,temproots}];
theroots=Transpose[DeleteCases[theroots,{_,{False}},1]][[1]];
If[theroots== {}[[1]],theroots={} ];

x=.
});
});
});
rmtranscend;

If[myfunc[[ival,1]]===y && myfunc[[jval,1]]===x,
theroots=N[{{myfunc[[jval,2]],(myfunc[[ival,2]]/.
x->myfunc[[jval,2]])}}];

If[myfunc[[ival,1]]===x && myfunc[[jval,1]]===y,
theroots=N[{{myfunc[[ival,2]],(myfunc[[jval,2]]/.
x->myfunc[[ival,2]])}}];

theroots=DeleteCases[DeleteCases[DeleteCases[
theroots, {x_Complex,y_}],{x_,y_Complex}],
{x_Complex,y_Complex}];

theroots=ReplaceAll[theroots, 0. -> 0];

If[theroots != {x,y} && theroots != {},{
endpoints=Join[endpoints, theroots] };

}];

If[Length[endpoints]>0,{
endpoints=Select[endpoints, (#[[1]]>=xlower)&];
endpoints=Select[endpoints, (#[[1]]<=xupper)&];
endpoints=Select[endpoints, (#[[2]]>=ylower)&];
endpoints=Select[endpoints, (#[[2]]<=yupper)&];
}];

```

```
]];
]];

intersect=Union[N[Round[endpoints*1000]/1000],{}];

If[Length[intersect]==1,
Print["The curves intersect at the point: ",f[intersect]]];

If[Length[intersect]>=2,{
Print["The curves intersect at the points:"];
Print[intersect] }];
```

III. FILE EXAMPLES.MA

(* Copyright Dennis A. Polaski, May 4,1993 *)

```
gr[1]:=Show[shade8,
Plot[ .4 Sin[x] +.2 ,(x,.2,1.25),
DisplayFunction -> Identity,
AxesLabel -> {"x","y"}, Ticks-> None,AspectRatio -> 1,
PlotRange -> {{0,1.35},{-.07,.6}}],
Graphics[{
Text["a", {.2,-.05}],
Text["b", {1.25,-.05}],
Text["y = f(x)", {.75,.6}]
}],
DisplayFunction -> $DisplayFunction];

example[1]:=Show [
Plot[x^2, {x,0,1},
DisplayFunction -> Identity,
AxesLabel -> {"x","y"},
Ticks -> {{1},{1}},
AspectRatio -> 1,
PlotRange -> {{-.06,1.1},{-.1,1}}],
Graphics[{
Line [{{1,0}, {1,1}}],
Text["0",{.04,-.08}],
Text ["y = x", {.5,.8}],
Text ["2",{.615,.85}]
}],
DisplayFunction -> $DisplayFunction];

example[2]:=Show[
Plot[ Cos[x], {x,-Pi/2, Pi/2},
DisplayFunction -> Identity,
AxesLabel -> {"x","y"},
AspectRatio->.5,
Ticks -> None,
PlotRange -> { {-1.65,1.65}, {-0.3, 1.2}}],
Graphics[{
Text[ " y = cos x", {1.4,1.05}],
Text[ "-Pi/2", {-Pi/2,-.2}],
Text[ "Pi/2", {Pi/2,-.2}]
}],
DisplayFunction -> $DisplayFunction];

gr[2]:=Show[
Plot[.5*(x-1)^3 +.2 ,(x,.5,1.5), DisplayFunction -> Identity,
AxesLabel -> {"x","y"}, Ticks-> None, AspectRatio ->1,
PlotRange -> {{0,2},{-.085,.265}}],
Graphics[{
```

```

Line[{{.5, 0}, {.5, (.5-1)^3+.325}},
Line[{{1.5, 0}, {1.5, .27}}],
Line[{{1.68, 0}, {1.68, .088}}],
Line[{{1.68, .13}, {1.68, .2}}],
Line[{{.5, .2}, {1.5, .2}}],
Line[{{1.58, .2}, {1.78, .2}}],
Line[{{.5, -.08}, {.5, -.0525}}],
Line[{{1.5, -.08}, {1.5, -.0525}}],
PointSize[.01], Table[Point[{1, n .005}], {n, 0, 39}],
Text["a", {.5, -.025}],
Text["b", {1.5, -.025}],
Text["c", {1, -.025}],
Text[" f(c)", {1.68, .11}],
Text["0", {-.075, -.025}],
Text["<--- (b - a) --->", {1.005, -.0675}],
Text["y = f(x)", {1, .25}],
Text[" √ ", {1.68, .01}],
Text[" ∧ ", {1.68, .19}]
}],
DisplayFunction -> $DisplayFunction];

```

```

example[3]:=Show[Plot[{Sqrt[4-x^2], -Sqrt[4-x^2]}, {x, -2, 2},
DisplayFunction -> Identity,
Ticks -> {{-2, 0, 2}, {}},
AxesLabel -> {"x", "y"},
AspectRatio -> Automatic,
PlotRange -> {{-2.15, 2.2}, {0, 2.1}},
Graphics[{
Text["y = 4 - x ", {2, 2.35}],
Text["2", {2.8, 2.65}]
}],
DisplayFunction -> $DisplayFunction];

```

```

example[4]:=Show[Plot[2x, {x, 1, 4},
DisplayFunction -> Identity,
Ticks -> {{1, 4}, {}},
AxesLabel -> {"x", "y"},
AspectRatio -> 1,
PlotRange -> {{0, 4.6}, {0, 8}},
Graphics[{
Text["y = 2x", {2, 7}],
Text["(1, 2)", {.5, 2}],
Text["(4, 8)", {4.6, 8}],
Line[{{1, 0}, {1, 2}}],
Line[{{4, 0}, {4, 8}}]
}],
DisplayFunction -> $DisplayFunction];

```

```

example[5]:=Show[Plot[{Cos[x], -Sin[x]}, {x, -.1, 1.6},
DisplayFunction -> Identity, AxesLabel -> {"x", "y"},
Ticks -> {{0}, {-1, 0, 1}}, AspectRatio -> Automatic,
PlotRange -> {{-.15, 1.8}, {-1.2, 1.1}},

```

```

Graphics[{
Line[{{Pi/2,0},{Pi/2,-1}}],
Text["Pi/2", {1.75,.2}],
Text["y = cos x", {1,1.1}],
Text["y = -sin x", {.75,-1.1}]
}],
DisplayFunction -> $DisplayFunction];

example[6]:=Show[shadel,Plot[1-x^(-2),(x,.5,5),
DisplayFunction -> Identity,
AxesLabel -> {"x","y"}, Ticks -> {{1,4},{1}},
AspectRatio -> Automatic,
PlotRange -> {{-.1,5},{-1.75,1.5}}],
Graphics[{
Line[{{0,1},{5.5,1}}],
Line[{{1,0},{1,1}}],
Line[{{4,15/16},{4,1}}],
Text["y = 1", {2,1.3}],
Text["y = 1 - x", {1.6,-1}],
Text["-2", {2.6,-.75}]
}],
DisplayFunction -> $DisplayFunction];

example[7]:=Show[Plot[{-x, 2-x^2),(x,-1,2),
DisplayFunction -> Identity, AxesLabel -> {"x","y"},
Ticks -> {{-1,0,1,2},{-2,0,2}}, AspectRatio -> Automatic,
PlotRange -> {{-1.95,2.2},{-2.4,2.1}}],
Graphics[{
Text[{"(-1,1)", {-1.65,1}],
Text[{"(2,-2)", {2,-2.2}],
Text["2", {2.35,1.9}],
Text["y = 2 - x ", {1.7,1.7}],
Text["y = - x", {.9,-1.8}]
}],
DisplayFunction -> $DisplayFunction];

example[8]:=Show[Plot[{{x^3+1,x^2+x},{x,-1.5,1.2},
DisplayFunction ->Identity, AxesLabel -> {"x","y"},
Ticks -> {{1},{2}}, AspectRatio -> Automatic,
PlotRange -> {{-1.75,1.5},{-.5,2.3}}],
Graphics[{
Text["y = x + 1",{-.7,1.2}],
Text["3",{-.625,1.35}],
Text["2",{1.175,.65}],
Text["y = x + x",{1.1,.5}],
Text[{"(-1,0)",{-1.5,-.2}],
Text[{"(1,2)",{1.4,2}]
}],
DisplayFunction -> $DisplayFunction];

example[9]:=Show[Plot[{{4+x^2,12-x^2),(x,-2.2,2.2},
DisplayFunction -> Identity, Ticks -> None,

```

```

AxesLabel -> {"x","y"}, AspectRatio -> Automatic,
PlotRange -> {{-4.5,4.5}, {2,13}},
Graphics[{
Text["(-2,8)",{-3.2,8}],
Text["(2,8)",{3.1,8}],
Text["y = 12 - x",{-3,12.5}],
Text["2",{-1.125,12.95}],
Text["y = 4 + x", {3,3.5}],
Text["2",{4.725,3.95}]
}],
DisplayFunction -> $DisplayFunction];

example[10]:=Show[Plot[Sqrt[x],{x,0,4},
DisplayFunction -> Identity,
AxesLabel -> {"x","y"}, Ticks -> {{2,4},{1,2}},
AspectRatio -> .75,
PlotRange -> {{-.2,4.5},{-.2,2.2}}],
Graphics[{
Line[{{2,0},{2,Sqrt[2]}}],
Line[{{2,0},{4,2}}],
Text["2",{1.7,2.2}],
Text["y = x", {1.85,2.1}],
Text["(4,2)",{4.4,2}],
Text["y = 0", {1,-.15}],
Text["y = x - 2", {4,.9}]
}],
DisplayFunction -> $DisplayFunction];

example[11]:=Show[shade2, Plot[2-x^2,{x,0,Sqrt[2]},
DisplayFunction -> Identity,
AxesLabel -> {"x","y"},
Ticks -> {{1},{1,2}}, AspectRatio -> Automatic,
PlotRange -> {{0,2},{-.07,2.2}}],
Graphics[{
Text["y = 2 - x", {.65,2.1}],
Text["2",{1,2.2}],
Text["y = x", {1.65,1.25}],
Text["y = 0",{.55,-.1}],
Line[{{0,0},{1.7,1.7}}],
Line[{{2.1,2.1},{2.3,1.55}}],
Line[{{1,0},{1,1}}]
}],
DisplayFunction -> $DisplayFunction];

example[12]:=Show[shade3,Plot[{Sqrt[x],-Sqrt[x],6-x},
{x,-.1,11},
DisplayFunction -> Identity, AspectRatio -> Automatic,
AxesLabel -> {"x","y"}, Ticks -> None,
PlotRange -> {{-.1,11},{-4.2,4.2}}],
Graphics[{
Line[{{0,0},{9,0}}],
Text["(9,-3)",{10,-2.2}],

```

```

Text["(4,2)",{4.2,3.2}],
Text["y = x", {2.2,-3}],
Text["2",{1.85,-2.67}],
Text["y = 6 - x",{7.2,1.1}]
}],
DisplayFunction -> $DisplayFunction];

gr[3]:=Show[Plot[4(x+.1)^3,{x,.2,.5},
DisplayFunction -> Identity,
AspectRatio -> .4, PlotRange -> {{-.1,1.2},{0,1}},
AxesOrigin -> {-.1,0},
AxesLabel -> {"x","y"},Ticks -> None],
Plot[4(x-.4)^3,{x,.7,1}, DisplayFunction -> Identity,
AspectRatio ->.5, PlotRange -> {{0,1},{-.1,1.3}}],
Graphics[{
Line[{{.485,.81},{.99,.81}}],
Line[{{.248,.175},{.756,.175}}],
Text["a",{-0.15,.175}],
Text["b",{-0.15,.81}],
Text["x = g(y)",{.2,.95}],
Text["x = f(y)",{1.1,.2}]
}], DisplayFunction -> $DisplayFunction];

gr[4]:=Show[Plot[2.1-(x+.3)^2,{x,.59,1.1},
DisplayFunction -> Identity,
AspectRatio ->.4,AxesLabel-> {"x","y"},Ticks -> None,
AxesOrigin -> {-.1,0},
PlotRange ->{{-.1,1.3},{0,1.4}}],
Plot[(x+.5)^3-.2,{x,.2,.66}, DisplayFunction -> Identity,
Ticks -> None,
AspectRatio ->.4, PlotRange -> {{0,1.3},{-.1,1.3}}],
Graphics[{Line[{{.25,.2},{1.08,.2}}],
Text["a",{-0.15,.2}],
Text["b",{-0.15,1.235}],
Text["x = f(y)",{1.2,.9}], Text["x = g(y)",{.2,1}]}],
DisplayFunction -> $DisplayFunction];

gr[5]:=Show[Plot[{-Sqrt[x^2-.5]+1.5,Sqrt[x^2-.5]+1.5},{x,.70
711,1.5},
DisplayFunction -> Identity, AspectRatio ->.5,
AxesLabel -> {"x","y"}, Ticks -> None,AxesOrigin -> {-.1,0},
PlotRange -> {{-.1,2.55},{0,3}}],
Plot[{-Sqrt[3-x^2]+1.5,Sqrt[3-x^2]+1.5},{x,1.2,1.732051},
DisplayFunction -> Identity, AspectRatio -> .5,
Ticks -> None],Graphics[{ Text["x = f(y)",{2.2,.5}],
Line[{{1.73205,1.45},{1.73205,1.55}}],
Text["a",{-0.25,.38}], Text["b",{-0.25,2.62}],
Text["x = g(y)",{.55,2.7}]}],
DisplayFunction -> $DisplayFunction];

```

```

example[13]:=Show[Plot[Sqrt[x],{x,0,4},
DisplayFunction -> Identity,
AxesLabel -> {"x","y"}, Ticks -> {{2,4},{1,2}},
AspectRatio -> Automatic,
PlotRange -> {{-.2,4.5},{-.45,2.2}}],
Graphics[{
Line[{{2,0},{4,2}}],
Text["2", {2.35, 2.3}],
Text["y = x", {1.8,2.1}],
Text["(4,2)",{4.4,2}],
Text["y = 0", {1,-.3}],
Text["x = y + 2", {4,.9}]
}],
DisplayFunction -> $DisplayFunction];

```

```

example[14]:=Show[shade2, Plot[2-x^2,{x,0,Sqrt[2]},
DisplayFunction -> Identity,
AxesLabel -> {"x","y"},
Ticks -> {{1},{1}}, AspectRatio -> Automatic,
PlotRange -> {{0,2.25},{-.22,2.05}}],
Graphics[{
Text["(1,1)",{1.3,1}],
Text["2",{1.5,.67}],
Text["x = 2 - y", {1.7,.6}],
Text["y = x", {1.4,.8}],
Text["y = 0", {.55,-.15}],
Line[{{0,0},{1.5,1.5}}]
}],
DisplayFunction -> $DisplayFunction];

```

```

example[15]:=Show[shade3,Plot[{Sqrt[x],-Sqrt[x],6-x},{x,-.1,
11},
DisplayFunction -> Identity, AspectRatio -> Automatic,
AxesLabel -> {"x","y"}, Ticks -> None,
PlotRange -> {{-.1,11},{-4.2,4.2}}],
Graphics[{
Line[{{0,0},{9,0}}],
Text["(9,-3)",{10,-2.2}],
Text["(4,2)",{4.2,3.2}],
Text["x = y",{2.8,-3.2}],
Text["2", {3.875,-2.8}],
Text["x = 6 - y",{7.5,.9}]
}],
DisplayFunction -> $DisplayFunction];

```

```

gr[6]:=Show[shade11,Plot[Sqrt[1-x^2],{x,.59,.975},
DisplayFunction -> Identity,
AxesLabel-> {"x","y"},AspectRatio -> Automatic,
Ticks -> None,PlotRange -> {{0,1.2},{0,.8}}],
Plot[Sqrt[.5-x^2],{x,.4375,.7071},DisplayFunction -> Identity,
Ticks -> None,AspectRatio -> 1],
Plot[Sqrt[.25-x^2],{x,.45,.5},DisplayFunction -> Identity,

```



```

Ticks -> None, AspectRatio -> 1],
Graphics[{
Text["r=f(Q)", {1.15, .255}],
Text["b", {.575, .5}],
Text["a", {.55, .125}],
Text["A", {.9, .55}],
Text["B", {.625, .865}],
Line[{{0,0},{1,.5}}],
Line[{{0,0},{.7,.88}}]
}],
DisplayFunction -> $DisplayFunction];

```

```

example[16]:=Show[PolarPlot
[2(1+Cos[Q]),{Q,0,2Pi}, DisplayFunction -> Identity,
AspectRatio -> Automatic, AxesLabel -> {"x","y"},
Ticks -> None, PlotRange -> {{-1,6},{-2.65,3}}],
Graphics[{
Line[{{0,0},{2.414, 2.414}}],
Line[{{4,0},{4.25,.225}}],
Text["2",{-2,2.1}],
Text["-2",{-2.25,-2.2}],
Text["Q=0,2Pi",{5.1,.35}],
Text["4", {4.2,-.25}],
Text["P(r,Q)", {2.6, 2.75}],
Text["r", {1.2, 1.45}],
Text["r= 2(1+cosQ)",{2.5,3.35}]
}],
DisplayFunction -> $DisplayFunction];

```

```

example[17]:=Show[shade12,PolarPlot[{1+Sin[Q]},{Q,0,2Pi},
DisplayFunction -> Identity, Ticks -> None,
AspectRatio -> Automatic,
AxesLabel -> {"x","y"}, PlotRange -> {{-1.8,2.5},{-1.1,2.3}}],
Graphics[{
Text["r = 1 + sinQ",{-0.95,2.2}],
Text["Q=Pi/4", {1.65,1.65}],
Text["Q=-Pi/4", {.85,-.85}],
Line[{{0,0},{1.4,1.4}}],
Line[{{0,0},{.7,-.7}}]
}],
DisplayFunction -> $DisplayFunction];

```

```

example[18]:=Show[shade7,PolarPlot[{2 Cos[Q]+1},{Q,0,2Pi},
DisplayFunction -> Identity, Ticks ->None,
AspectRatio -> Automatic, AxesLabel -> {"x","y"},
PlotRange-> {{-1.9,4},{-2,2.3}}],
Graphics[{
Line[{{-0.75,1.3},{.5,-.866}}],
Line[{{-0.75,-1.3},{.5,.866}}],
Line[{{3,0},{3.2,.3}}],
Line[{{1,0},{1.2,.3}}],
Line[{{0,0},{3,0}}],

```

```

Text["r = 2 cosQ + 1", {1.75, 2.15}],
Text["Q=2Pi/3", {-1.2, 1.45}],
Text["Q=4Pi/3", {-1.2, -1.45}],
Text["Q=Pi", {1.65, .4}],
Text["Q=0", {3.55, .4}]
}],
DisplayFunction -> $DisplayFunction];

```

```

example[19]:=Show[shade10, PolarPlot[{Sin[3Q]}, {Q, 0, 2Pi},
DisplayFunction -> Identity, Ticks -> None,
PlotRange -> {{-0.9, 1.2}, {-1.05, 1.1}},
AspectRatio -> Automatic, AxesLabel -> {"x", "y"}],
Plot[Sqrt[1.1-x^2], {x, .908, 1.05}, DisplayFunction -> Identity,
Ticks -> None, AspectRatio -> Automatic],
Graphics[{
Line[{{0, 0}, {1.039, .6}}],
Line[{{0, 0}, {.6, 1.039}}],
Text["r = sin3Q", {-0.6, .9}],
Text["Q=Pi/3", {.6, 1.15}],
Text["Q=Pi/6", {1.15, .725}],
Text["Q=0", {1, -.1}],
}],
DisplayFunction -> $DisplayFunction];

```

```

example[20]:=Show[PolarPlot[ {Sqrt[Cos[2Q]]}, {Q, 0, 2Pi},
DisplayFunction -> Identity, AxesLabel -> {"x", "y"},
Ticks -> None, AspectRatio -> Automatic,
PlotRange -> {{-1.1, 1.1}, {-0.8, 1.1}}],
Graphics[{
Line[{{-0.6, -0.6}, {0.6, 0.6}}],
Line[{{-0.6, 0.6}, {0.6, -0.6}}],
Text["r = cos2Q", {-0.4, .95}],
Text["2", {-0.6, 1.05}],
Text["Pi/4", {0.7, 0.7}],
Text["-Pi/4", {0.7, -0.7}],
Text["3Pi/4", {-0.75, 0.7}],
Text["5Pi/4", {-0.75, -0.7}],
Text["1", {1.05, -.1}]
}],
DisplayFunction -> $DisplayFunction];

```

```

gr[7]:=Show[shade6, Plot[Sqrt[1-x^2], {x, .555, .896},
DisplayFunction -> Identity, AxesLabel -> {"x", "y"},
Ticks -> None, AspectRatio -> Automatic],
Plot[.5x, {x, 0, 1.275}, DisplayFunction -> Identity,
Ticks -> None],
Plot[1.5x, {x, 0, .665}, DisplayFunction -> Identity,
Ticks -> None],
Plot[Sqrt[.75-x^2+x], {x, .658, 1.27},
DisplayFunction -> Identity, Ticks -> None],
Graphics[{
Text["r1", {.65, .65}],

```

```

Text["r2",{1,.95}],
Text["Q=a",{1.2,.475}],
Text["Q=b",{.45,.95}]
]],
DisplayFunction -> $DisplayFunction];

example[21]:=Show[shade5,shade4,
PolarPlot[1,{Q,0,2Pi},AxesLabel -> {"x","y"},
Ticks -> None, AspectRatio -> Automatic,
DisplayFunction -> Identity],
PolarPlot[.85,{Q,0,1.3},
Ticks -> None, AspectRatio -> Automatic,
DisplayFunction -> Identity],
PolarPlot[{1-Cos[Q]},{Q,0,2Pi}, DisplayFunction -> Identity,
Ticks -> None, AspectRatio -> Automatic],
Graphics[{
Line[{{0,-1},{.2,-1.3}}],
Line[{{0,1},{.2,1.3}}],
Line[{{0,0},{.2,.98}}],
Line[{{0,0},{.3,.954}}],
Line[{{0,0},{1,0}}],
Text["Q=-Pi/2",{.5,-1.4}],
Text["Q=Pi/2",{.4,1.4}],
Text["Q",{.6,.4}],
Text["r2 = 1",{.55,1.15}],
Text["r1 = 1 - cosQ",{-1.1,1.5}]
]],
DisplayFunction -> $DisplayFunction];

example[22]:=Show[shade9,PolarPlot[{3Cos[Q],
1+Cos[Q]},{Q,0,2Pi},
DisplayFunction -> Identity, AspectRatio -> Automatic,
AxesLabel -> {"x","y"}, Ticks -> None,
PlotRange -> {{-.3,3.75},{-2.5,2.5}}],
Graphics[{
Line[{{0,0},{1.25,2.16}}],
Line[{{0,0},{3,0}}],
Line[{{0,0},{1.25,-2.16}}],
Line[{{1.6,-1},{2.2,-1.9}}],
Line[{{1.9,1.8},{1.7,1.5}}],
Text["Q=Pi/3",{1.25,2.3}],
Text["Q=-Pi/3",{1.25,-2.3}],
Text["r2 = 3cosQ",{2.7,1.95}],
Text["r1 = 1 + cosQ",{3.05,-2.05}]
]],
DisplayFunction -> $DisplayFunction];

example[23]:=Show[PolarPlot[{1-Cos[Q]},{Q,0,2Pi},
DisplayFunction->Identity,
Ticks->None, AxesLabel->{"x","y"},AspectRatio->Automatic,
PlotRange->{{-2.75,1},{-1.3,1.3}}],
Plot[Sqrt[.1-x^2],{x,-.161,.318},DisplayFunction->Identity,

```

```
Ticks-> None, AspectRatio->Automatic],
Graphics[{
Line[{{0,0},{-.75,1.3}}],
Text["Q",{.35,.2}],
Text["r",{-.55,.55}],
Text["r = 1-cosQ",{-2.2,1.2}],
Text["P(r,Q)",{-.85,1.45}],
Text["-2",{-2.15,-.15}],
Text["1",{-.1,1.2}]
}],
DisplayFunction->$DisplayFunction];
```

IV. FILE INTEGRAL.MA

(* Copyright Dennis A. Polaski, May 4, 1993 *)

```
BeginPackage["integral`"];
```

```
graph::usage = "Type graph, to invoke a program which  
plots a region, finds the limits of integration, and  
evaluates the integral."
```

```
SetOptions[Plot, AxesLabel -> {"x", "y"}];  
SetOptions[ParametricPlot, PlotPoints -> 600];  
SetOptions[Plot, PlotPoints->150];
```

```
Off[Drop::drop];  
Off[EndPackage::noctx];  
Off[FindRoot::cvnwt];  
Off[General::spell];  
Off[General::spell1];  
Off[General::stop];  
Off[Get::noopen];  
Off[Graphics::gprim];  
Off[Graphics::gpfn];  
Off[Greater::nord];  
Off[Infinity::indet];  
Off[NIntegrate::ncvb];  
Off[NIntegrate::nlim];  
Off[NIntegrate::slwcon];  
Off[ParametricPlot::pptr];  
Off[Part::partd];  
Off[Part::partw];  
Off[Plot::plln];  
Off[Plot::plnr];  
Off[Power::infy];  
Off[Remove::remal];  
Off[ReplaceAll::rmix];  
Off[ReplaceAll::reps];  
Off[Show::gtype];  
Off[Solve::ifun];  
Off[Solve::tdep];  
Off[Unset::norep];
```

```
Needs["Graphics`Master`"];  
<<c:polar1.ma;  
<<c:polar2.ma;  
<<c:cartplot.ma;  
<<c:inverse.ma;
```

```

<<c:examples.ma;
<<c:shades.ma;
<<c:cart.ma;
<<c:polplot1.ma;
<<c:polplot2.ma;

Begin[``Private`"];

graph:=
Module[{} ,

labexample=Input["To display a graph that is in the lab
book type in the example number, otherwise type n."];

If[labexample != n, example[labexample], {

type2D=Input["Type 1 for cartesian coordinate system, or
type 2 for polar coordinate system."];

If[type2D==1, {
plotonly=Input["To just plot curves type y, otherwise
type n."];
If[plotonly===y, cartplot, cart]
}];

If[type2D==2, {
plotonly=Input["To just plot curves type y, otherwise
type n."];
If[plotonly===y, {
typepolplot=Input["Type 1 for plot of area between the
origin and a polar curve, or type 2 for plot of area
between two polar curves."];
If[typepolplot==1, polplot1];
If[typepolplot==2, polplot2]}, {
typepolar=Input["Type 1 for area between the origin and
a polar curve, or type 2 for area between two polar curves."];
If[typepolar==1, polar1];
If[typepolar==2, polar2];
}];
}];
}];
]

End[];

EndPackage[];

```

V. FILE INVERSE.MA

(* Copyright Dennis A. Polaski, May 4,1993 *)

```

inverse:=Module[{}],
ln[x_]:=Log[x];
If[myfunc==(myfunc /. Log[x_]->x),{

transcend;
f[{x___}]:= x;
allycoord=Transpose[Join[f[thelines]]][[2]];
ycoord=Union[ycoord, {Max[allycoord]}, {Min[allycoord]}];
ycoord=N[Union[Round[ycoord*100]/100];
ycoord=Union[ReplaceAll[ycoord, 0.->0],{}];
xfuncs=Transpose[Partition[Join[myfunc, thelines],
Length[myfunc]]];

yfunc=Complement[xfuncs,Cases[xfuncs, {y==b_Integer,c_} |
{y==b_Real,c_}]];

For[ww=1, ww<=Length[yfunc],{wval=ww; ++ww;
If[yfunc[[wval,1,1]]==y,
yfunc[[wval,1]]=ToExpression[StringJoin["x==",
"(x /. Solve[yfunc[[wval,1,2]]==y, x,
InverseFunctions-> True]][[1]]"] ]];
If[yfunc[[wval,1]]==True |
(yfunc[[wval,1,2]] /. (y___)^(1/3)->y)!=yfunc[[wval,1,2]],
ww=99]
}];

If[ww<99, {
revlines=Transpose[yfunc][[2]];
For[vv=1, vv<=Length[ycoord]-1, {vval=vv; ++vv;
yy=(ycoord[[vval]]+ycoord[[vval+1]])/2;
integrfunc=Select[revlines,((yy> Min[Transpose[#] [[2]])] &&
(yy< Max[Transpose[#] [[2]])]&];
If[Length[integrfunc]>2,{vv=99; ww=99}];
}];
}];

If[ww<99, {
Pause[8];
yintegr=Input["Type y if you wish to integrate the same
problem with respect to y, otherwise type n."];

If[yintegr==y,{
Print[" "];
Print["*****"];
invfunc=Transpose[yfunc][[1]];
tempinvfunc=invfunc;
revlines=Transpose[yfunc][[2]];

```

```

For[zz=1, zz<=Length[ycoord]-1, {zval=z; ++z;
yy=(ycoord[[zval]]+ycoord[[zval+1]])/2;
integrfunc=Select[revlines, ((yy> Min[Transpose[#] [[2]]) &&
(yy< Max[Transpose[#] [[2]]]))&];

invfunc=tempinvfunc;

If[Length[integrfunc]==2, {
c1=Drop[Select[integrfunc[[1]] ,
#[[2]]>=ycoord[[zval]] && #[[2]]<=ycoord[[zval+1]]&], 1];
c2=Drop[Select[integrfunc[[2]] ,
#[[2]]>=ycoord[[zval]] && #[[2]]<=ycoord[[zval+1]]&], -1];
avgcurve1=(Transpose[c1][[1,1]]+Transpose[c1][[1,-1]])/2;
avgcurve2=(Transpose[c2][[1,1]]+Transpose[c2][[1,-1]])/2;

adjc1=Transpose[Reverse[Transpose[c1]]];
maxc1=Max[Union[{adjc1[[-1,1]]}, {adjc1[[1,1]]}]];
minc1=Min[Union[{adjc1[[-1,1]]}, {adjc1[[1,1]]}]];
curve1=NIntegrate[Fit[adjc1, {1,y,y^2}, y],
{y, minc1, maxc1}]/(maxc1-minc1);
adjc2=Transpose[Reverse[Transpose[c2]]];
maxc2=Max[Union[{adjc2[[-1,1]]}, {adjc2[[1,1]]}]];
minc2=Min[Union[{adjc2[[-1,1]]}, {adjc2[[1,1]]}]];
curve2=NIntegrate[Fit[adjc2, {1,y,y^2}, y],
{y, minc2, maxc2}]/(maxc2-minc2);
}];

If[Length[integrfunc]==2, {
If[avgcurve2>avgcurve1, {integrfunc=Reverse[integrfunc];
tempcurve=curve2; curve2=curve1; curve1=tempcurve;
tempcurve=avgcurve2; avgcurve2=avgcurve1; avgcurve1=tempcurve} ]
}];

If[Length[integrfunc]==2, {
place1=Position[revlines, integrfunc[[1]][[1]][[1]]];
place2=Position[revlines, integrfunc[[2]][[1]][[1]]];
{theplace=Position[revlines, integrfunc[[1]][[1]][[1]]]};

If[Length[integrfunc]==2, {
If[Replace[invfunc[[place1,2]], Sqrt[x_]->0] !=
invfunc[[place1,2]],
If[curve1<avgcurve1,
invfunc[[place1,2]]=negbranch[invfunc[[place1,2]] ] ]
}];
If[Length[integrfunc]==2, {
If[Replace[invfunc[[place2,2]], Sqrt[x_]->0] !=
invfunc[[place2,2]],
If[curve2<avgcurve2,
invfunc[[place2,2]]=negbranch[invfunc[[place2,2]] ] ]
}];
}];

```



```

y=.;
If[Length[integrfunc]==2, {
area=NIntegrate[invfunc[[place1,2]]-invfunc[[place2,2]],
(y, ycoord[[zval]],ycoord[[zval+1]])],{
area=NIntegrate[Evaluate[2bothbranches[invfunc[[theplace,2]]
]],
(y,ycoord[[zval]], ycoord[[zval+1]])]}}];

If[Length[integrfunc]==1, {
right=invfunc[[theplace,2]];
left=negbranch[invfunc[[theplace,2]] ]},
(right=invfunc[[place1,2]];left=invfunc[[place2,2]]]};

Print[" "];
If[zval==1 && Length[ycoord]==2, {
Print["The limits of integration are y = ",
ycoord[[zval]]," and y = ", ycoord[[zval+1]] ];
Print["The right curve is x = ", right];
Print["The left curve is x = ", left];
Print["The area of the region is ", Abs[area] ] ]};

If[zval==1 && Length[ycoord]>=3, {
Print["The limits of integration for the first region"];
Print["are y = ",ycoord[[zval]],
" and y = ",ycoord[[zval+1]]];
Print["The right curve for the first region is x = ", right];
Print["The left curve for the first region is x = ", left];
Print["The area of the first region is ", Abs[area] ];
areal=Abs[area] ]};

If[zval==2 && Length[ycoord]>=3, {
Print["The limits of integration for the second region"];
Print["are y = ",ycoord[[zval]],
" and y = ",ycoord[[zval+1]]];
Print["The right curve for the second region is x = ", right];
Print["The left curve for the second region is x = ", left];
Print["The area of the second region is ", Abs[area] ];
If[Length[ycoord]==3,
Print["The area of the total region is ", areal+Abs[area] ],
area2=Abs[area] ]
}];

If[zval==3 && Length[ycoord]>=4, {
Print["The limits of integration for the third region"];
Print["are y = ",ycoord[[zval]],
" and y = ",ycoord[[zval+1]]];
Print["The right curve for the third region is x = ", right];
Print["The left curve for the third region is x = ", left];
Print["The area of the third region is ", Abs[area] ];
If[Length[ycoord]==4,
Print["The area of the total region is ", areal+area2+
Abs[area]], area3=Abs[area] ]
}];

```

```

]];
If[zval==4 && Length[ycoord]==5, (
Print["The limits of integration for the fourth region"];
Print["are y = ", ycoord[[zval]],
" and y = ", ycoord[[zval+1]]];
Print["The right curve for the fourth region is x = ", right];
Print["The left curve for the fourth region is x = ", left];
Print["The area of the fourth region is ", Abs[area] ];
Print["The area of the total region is ", areal+area2+area3+
Abs[area]]
)];
)];
)];
)];
)];
)];
]

```

```

g1[(b_ + Sqrt[a_])*c_] := (b - Sqrt[a])*c;
g2[(b_ + d_*Sqrt[a_])*c_] := (b - d Sqrt[a])*c;
g3[ b_ + d_*Sqrt[a_] ] := b - d Sqrt[a];
g4[ Sqrt[a_] *c_] := -Sqrt[a] *c;
g5[ Sqrt[a_] ] := -Sqrt[a];
g6[ b_ + Sqrt[a_] ] := b - Sqrt[a];

```

```

h1[(b_ + Sqrt[a_])*c_] := Sqrt[a]*c;
h2[(b_ + d_*Sqrt[a_])*c_] := d*Sqrt[a]*c;
h3[ b_ + d_*Sqrt[a_] ] := d*Sqrt[a];
h4[ Sqrt[a_] *c_] := Sqrt[a]*c;
h5[ Sqrt[a_] ] := Sqrt[a];
h6[ b_ + Sqrt[a_] ] := Sqrt[a];

```

```

negbranch[x_] :=
If[Cases[{x}, (b_ + Sqrt[a_])*c_] != {}, g1[x],
If[Cases[{x}, (b_ + d_*Sqrt[a_])*c_] != {}, g2[x],
If[Cases[{x}, b_ + d_*Sqrt[a_] ] != {}, g3[x],
If[Cases[{x}, Sqrt[a_] *c_] != {}, g4[x],
If[Cases[{x}, Sqrt[a_] ] != {}, g5[x],
If[Cases[{x}, b_ + Sqrt[a_] ] != {}, g6[x]]]]]]];

```

```

bothbranches[x_] :=
If[Cases[{x}, (b_ + Sqrt[a_])*c_] != {}, h1[x],
If[Cases[{x}, (b_ + d_*Sqrt[a_])*c_] != {}, h2[x],
If[Cases[{x}, b_ + d_*Sqrt[a_] ] != {}, h3[x],
If[Cases[{x}, Sqrt[a_] *c_] != {}, h4[x],
If[Cases[{x}, Sqrt[a_] ] != {}, h5[x],
If[Cases[{x}, b_ + Sqrt[a_] ] != {}, h6[x]]]]]]];

```

VI. FILE POLAR1.MA

(* Copyright Dennis A. Polaski, May 4, 1993 *)

```

polar1:=
Module[{
    func, limitsknown, theroots, lowerlimit,
    upperlimit, newcurve, area, totalarea={}, regions={},
    curvesknown, func1, func2, temp
},
Q=.;
cltranscend;
func=Input["Enter the polar curve which bounds the region.

r = "];
func=func /. sqrt->Sqrt;
func=func /. pi->Pi;
func=func /. q->Q;
revfunc=Replace[func, Sqrt[x_] -> x];
limitsknown=Input["If the limits are known type y,
otherwise type n."];

If[limitsknown===y,
{lowerlimit=Input["Input lower limit."];
lowerlimit=lowerlimit /. sqrt->Sqrt;
lowerlimit=lowerlimit /. pi->Pi;
While[lowerlimit < -N[Pi], {lowerlimit=Input["The lowerlimit
can not be less than -Pi, input lower limit again."];
lowerlimit=lowerlimit /. sqrt->Sqrt;
lowerlimit=lowerlimit /. pi->Pi;
}];
upperlimit=Input["Input upper limit."];
upperlimit=upperlimit /. sqrt->Sqrt;
upperlimit=upperlimit /. pi->Pi;
While[upperlimit > 2 N[Pi], {upperlimit=Input["The upperlimit
can not be greater than 2Pi, input upper limit again."];
upperlimit=upperlimit /. sqrt->Sqrt;
upperlimit=upperlimit /. pi->Pi;
}];
If[N[upperlimit] < N[lowerlimit], lowerlimit=N[lowerlimit-2Pi]]
},
(lowerlimit=0.; upperlimit= 2Pi)];

rmtranscend;
max=Max[DeleteCases[Table[func /. Q -> values,
(values, 0, 2Pi, .25)], x_Complex]];

ray1=N[(func /. Q->lowerlimit)];
ray2=N[(func /. Q->upperlimit)];

```

```

llequivalent=N[Mod[lowerlimit, 2Pi]];
ulequivalent=N[Mod[upperlimit, 2Pi]];

If[lowerlimit < 0, llequivalent= 2 N[Pi]+lowerlimit,
llequivalent= lowerlimit];
If[upperlimit < 0, ulequivalent= 2 N[Pi]+upperlimit,
ulequivalent= upperlimit];

Q=.;
If[(ray1==ray2) && (llequivalent==ulequivalent) ||
(ray1==0 && ray2==0),{
PolarPlot[func,{Q, topi[lowerlimit], topi[upperlimit]},
AxesLabel -> {"x","y"}, AspectRatio -> Automatic,
PlotRange-> {{-1.02 max, 1.02 max},{-1.02 max, 1.02 max}}],
{
Show[PolarPlot[func,{Q, topi[lowerlimit], topi[upperlimit]},
DisplayFunction -> Identity, AxesLabel -> {"x","y"},
AspectRatio -> Automatic, PlotRange-> {{-1.02 max,1.02 max},
{-1.02 max, 1.02 max}}],
Graphics[{
Line[{{0,0},{Cos[lowerlimit]*ray1, Sin[lowerlimit]*ray1}],
Line[{{0,0},{Cos[upperlimit]*ray2, Sin[upperlimit]*ray2}],
}],
DisplayFunction-> $DisplayFunction}}];

ptranscend;
theroots=NSolve[{r==0, r==revfunc},{r,Q}];
theroots=N[{r,Q} /. theroots];

If[theroots!={r,Q} && theroots !={},{
theroots>DeleteCases[theroots,{r_,Q_Complex /;
Abs[Im[Q]]>.1}];
theroots=ReplaceAll[theroots,{r_,Q_} -> {r,Re[Q]}];
rmtranscend;
Q=.;
theroots=Table[FindRoot[revfunc==0,
{Q,theroots[[m,2]}]],{m,Length[theroots]}];
theroots={0,Q} /. theroots;
theroots=N[Union[theroots,{}]];
theroots=Union[theroots, theroots+ N[Table[{0,2Pi},
{i,Length[theroots]}]]];
theroots=Select[theroots,({#[[2]]<=N[upperlimit] &&
#[[2]] >= N[lowerlimit]}&);
theroots=Union[N[(Round[100*theroots])/100],{}];
}];
rmtranscend;

If[limitsknown!=y, {
If[theroots!={r,Q} && theroots!={} &&
Length[theroots]>=1,{
Print[" "];

```

```

Print["The polar curve r is equal to zero when Q is:
      ",Map[topi,Transpose[theroots][[2]]] ]
]];

Pause[8];
newcurve=Input["The polar curve is plotted from Q=0
to Q=2 Pi. If you want to change the limits type y, otherwise
type n."];

If[newcurve===y,{
lowerlimit=Input["Input new lower limit."];
lowerlimit=lowerlimit /. sqrt->Sqrt;
lowerlimit=lowerlimit /. pi->Pi;
While[lowerlimit <-N[Pi], {lowerlimit=Input["The lower limit
can not be less than -Pi, input lower limit again."];
lowerlimit=lowerlimit /. sqrt->Sqrt;
lowerlimit=lowerlimit /. pi->Pi;
}]];
upperlimit=Input["Input new upper limit."];
upperlimit=upperlimit /. sqrt->Sqrt;
upperlimit=upperlimit /. pi->Pi;
While[upperlimit>2 N[Pi], {upperlimit=Input["The upperlimit
can not be greater than 2Pi, input upper limit again."];
upperlimit=upperlimit /. sqrt->Sqrt;
upperlimit=upperlimit /. pi->Pi;
}]];
If[N[upperlimit]<N[lowerlimit],lowerlimit=N[lowerlimit-2Pi]];

ray1=N[(func /. Q->lowerlimit)];
ray2=N[(func /. Q->upperlimit)];

llequivalent=N[Mod[lowerlimit, 2Pi]];
ulequivalent=N[Mod[upperlimit, 2Pi]];

If[lowerlimit < 0, llequivalent= 2 N[Pi]+lowerlimit,
llequivalent= lowerlimit];
If[upperlimit < 0, ulequivalent= 2 N[Pi]+upperlimit,
ulequivalent= upperlimit];

Q=.;
If[(ray1==ray2) && (llequivalent==ulequivalent)||
(ray1==0 && ray2==0),{
PolarPlot[func,{Q, topi[lowerlimit], topi[upperlimit]},
AxesLabel -> {"x","y"}, AspectRatio -> Automatic,
PlotRange-> {{-1.02 max, 1.02 max},{-1.02 max, 1.02 max}}],
{
Show[PolarPlot[func,{Q, topi[lowerlimit], topi[upperlimit]},
DisplayFunction -> Identity, AxesLabel -> {"x","y"},
AspectRatio -> Automatic, PlotRange-> {{-1.02 max,1.02 max},
{-1.02 max, 1.02 max}}],
Graphics[{
Line[{{0,0},{Cos[lowerlimit]*ray1, Sin[lowerlimit]*ray1}]],

```

```

Line[{{0,0},{Cos[upperlimit]*ray2, Sin[upperlimit]*ray2}},
  ]],
DisplayFunction-> $DisplayFunction]]]
  ]]
  ]];

ptranscend;
theroots=NSolve[{r==0, r==revfunc},{r,Q}];
theroots=N[{r,Q} /. theroots];

If[theroots!==(r,Q) && theroots !={},{
theroots=DeleteCases[theroots,{r_,Q_Complex /;
Abs[Im[Q]]>.1}];
theroots=ReplaceAll[theroots,{r_,Q_} -> {r,Re[Q]}];
rmtranscend;
Q=.;
theroots=Table[FindRoot[revfunc==0,
{Q,theroots[[m,2]}]],{m,Length[theroots]}];
theroots={0,Q} /. theroots;
theroots=N[Union[theroots,{}]];
theroots=Union[theroots,theroots+ N[Table[{0,2Pi},
{i,Length[theroots]}]]];
theroots=Select[theroots,#[[2]]<=N[upperlimit] &&
#[[2]] >= N[lowerlimit]&];
theroots=Union[N[(Round[100*theroots])/100],{}];
]];

If[theroots =={(r,Q)}||Length[theroots]<=1, {
area=NIntegrate[.5*(func)^2, {Q,topi[lowerlimit],
topi[upperlimit]}];
cltranscend;
Print[" "];
Print["The limits of integration for the region bounded by
the"];
Print["polar curve r = ",func, ", and the origin are:"];
Print[" "];
Print["          a=",topi[Chop[lowerlimit,.01]]," and b=",
topi[Chop[upperlimit,.01]],"."];
Print[" "];
Print["The area bounded by the polar curve and the origin is
",
N[area,3]]];

If[theroots !={(r,Q) && theroots !={} && Length[theroots]>1, {
theroots=Transpose[theroots];
theroots=theroots[[2]];
theroots=Sort[Union[ReplaceAll[Join[
{N[(Round[100*lowerlimit])/100]},
theroots,{N[(Round[100*upperlimit])/100]}],0.->0},{}]];

```

```

rmtranscend;
For[i=1, i<=Length[theroots]-1, {ival=i; ++i; area=.;
Q=(theroots[[ival]] + theroots[[ival+1]])/2;
If[func!=revfunc, {
If[N[func] >0, { Q=.;
area=NIntegrate[.5*(func)^2, {Q, topi[theroots[[ival]] ],
topi[theroots[[ival+1]] ]}];
AppendTo[totalarea, Abs[area]];
AppendTo[regions, ival] }}
]];
If[func==revfunc, {
If[Abs[N[func]] >0, { Q=.;
area=NIntegrate[.5*(func)^2, {Q, topi[theroots[[ival]] ],
topi[theroots[[ival+1]] ]}];
AppendTo[totalarea, Abs[area]];
AppendTo[regions, ival] }}
]];

area=Apply[Plus, totalarea];
numregions=Length[regions];
cltranscend;
If[numregions==1, {Q=.;
Print[" "];
Print["The limits of integration for the region bounded by
the"];
Print["polar curve r = ", func," and the origin are:"];
Print[" "];
Print["          a=",topi[Chop[theroots[[ regions[[1]]
]],.01]],
" and b=",topi[Chop[theroots[[regions[[1]] + 1]],.01]],"." ];
Print[" "];
Print["The area bounded by the polar curve and the origin
is ",N[area,3] ]}}];

If[numregions >1, {
Print[" "];
Print["The limits of integration for the regions which
are bounded"]; Q=.; cltranscend;
Print["by the polar curve r = ", func," and the
origin are:"]; Print[" "];

If[Abs[N[Mod[theroots[[regions[[1]] ]]+2Pi,2Pi]] -
N[Mod[theroots[[regions[[numregions]]+1]]+2Pi,2Pi]]]<.1,{

If[theroots[[regions[[1]] ]]<0 &&
theroots[[regions[[numregions]] ]]>=N[Pi],
Print["          a1=",
topi[Chop[N[theroots[[ regions[[ numregions ]
]]-2Pi],.01]],"b1=",
topi[Chop[theroots[[regions[[1]]+1]],.01]] ]}];

```

```

If[theroots[[regions[[1]] ]]<0 &&
theroots[[regions[[numregions]] ]]<=N[Pi],
Print["          a1=",
topi[Chop[theroots[[regions[[numregions]] ]],.01]],
" and b1=",topi[Chop[N[theroots[[regions[[1]]+1]]+2Pi],.01]]
]];

If[theroots[[regions[[1]] ]]>=0 &&
theroots[[regions[[numregions]] ]]>0,
Print["          a1=",
topi[Chop[N[theroots[[regions[[numregions]] ]]-2Pi],.01]],
" and b1=",topi[Chop[theroots[[regions[[1]]+1]],.01]] ]];

For[j=3,j<=numregions, {jval=j; ++j;
Print["          a",jval-1,"=",
topi[Chop[theroots[[regions[[jval-1]] ]],.01]],
" and b",jval-1,"=",
topi[Chop[theroots[[regions[[jval-1]]+1]],.01]] ]}]
),

{For[j=1, j<=numregions,{jval=j; ++j;
Print["          a", jval,"=",
topi[Chop[theroots[[ regions[[jval]] ]],.01]],
" and b",jval,"=",
topi[Chop[theroots[[ regions[[jval]]+1 ]],.01]] ] }];
}];
Print[" "]; Q=.;
Print["The total area bounded by the polar curve and the
origin is"];
Print[" ",N[area,3]];
}];
}];
]

```

```

ptranscend:=Module[{},
sin[x_]:= x - x^3/6 + x^5/120 - x^7/5040 + x^9/362880 -
x^11/39916800+x^13/6227020800-x^15/1307674368000+
x^17/355687428096000-x^19/121645100408832000;
x^21/51090942171709440000 -
x^23/25852016738884976640000 +
x^25/15511210043330985984000000;
cos[x_]:= 1 - x^2/2 + x^4/24 - x^6/720 + x^8/40320 -
x^10/3628800 + x^12/479001600;
tan[x_]:= x + x^3/3 + 2x^5/15 + 17x^7/315 + 62x^9/2835 +
1382x^11/155925;
cot[x_]:= 1/x -x/3 -x^3/45 - 2x^5/945 - x^7/4725 -
2x^9/93555;
csc[x_]:= 1/x + x/6 + 7x^3/360 + 31x^5/15120 +
127x^7/604800 + 73x^9/3421440;
sec[x_]:= 1 + x^2/2 + 5x^4/24 + 61x^6/720 + 277x^8/8064 +
50521x^10/3628800;

```



```
exp[x_] := 1 + x + x^2/2 + x^3/6 + x^4/24 + x^5/120 + x^6/720 +  
x^7/5040 + x^8/40320 + x^9/362880]
```

```
cltranscend := Module[(), sin[x_] =.; cos[x_] =.; tan[x_] =.;  
cot[x_] =.; csc[x_] =.; sec[x_] =.; exp[x_] =.]
```

```
rmtranscend := Module[(), sin[x_] := Sin[x]; cos[x_] := Cos[x];  
tan[x_] := Tan[x]; cot[x_] := Cot[x]; csc[x_] := Csc[x];  
sec[x_] := Sec[x]; exp[x_] := Exp[x] ]
```

```
topi[-.26] := -Pi/12; topi[-.28] := -Pi/11; topi[-.29] := -Pi/11;  
topi[-.31] := -Pi/10; topi[-.34] := -Pi/9; topi[-.35] := -Pi/9;  
topi[-.39] := -Pi/8; topi[-.44] := -Pi/7; topi[-.45] := -Pi/7;  
topi[-.52] := -Pi/6; topi[-.57] := -2Pi/11; topi[-.62] := -Pi/5;  
topi[-.63] := -Pi/5; topi[-.69] := -2Pi/9; topi[-.70] := -2Pi/9;  
topi[-.78] := -Pi/4; topi[-.79] := -Pi/4; topi[-.85] := -3Pi/11;  
topi[-.86] := -3Pi/11; topi[-.94] := -3Pi/10; topi[-1.04] := -Pi/3;  
topi[-1.05] := -Pi/3; topi[-1.14] := -4Pi/11; topi[-1.17] := -3Pi/8;  
topi[-1.18] := -3Pi/8; topi[-1.25] := -2Pi/5; topi[-1.26] := -2Pi/5;  
topi[-1.30] := -5Pi/12; topi[-1.31] := -5Pi/12;  
topi[-1.34] := -3Pi/7; topi[-1.35] := -3Pi/7; topi[-1.39] := -4Pi/9;  
topi[-1.40] := -4Pi/9; topi[-1.42] := -5Pi/11;  
topi[-1.43] := -5Pi/11; topi[-1.57] := -Pi/2;  
topi[-1.71] := -6Pi/11; topi[-1.74] := -5Pi/9;  
topi[-1.75] := -5Pi/9; topi[-1.79] := -4Pi/7; topi[-1.80] := -4Pi/7;  
topi[-1.83] := -7Pi/12;  
topi[-1.88] := -3Pi/5; topi[-1.96] := -5Pi/8;  
topi[-1.99] := -7Pi/11; topi[-2.00] := -7Pi/11;  
topi[-2.09] := -2Pi/3; topi[-2.19] := -7Pi/10;  
topi[-2.20] := -7Pi/10; topi[-2.24] := -5Pi/7;  
topi[-2.28] := -8Pi/11; topi[-2.35] := -3Pi/4;  
topi[-2.36] := -3Pi/4; topi[-2.44] := -7Pi/9; topi[-2.51] := -4Pi/5;  
topi[-2.57] := -9Pi/11; topi[-2.61] := -5Pi/6;  
topi[-2.62] := -5Pi/6; topi[-2.69] := -6Pi/7; topi[-2.74] := -7Pi/8;  
topi[-2.75] := -7Pi/8; topi[-2.79] := -8Pi/9;  
topi[-2.82] := -9Pi/10; topi[-2.83] := -9Pi/10;  
topi[-2.85] := -10Pi/11; topi[-2.86] := -10Pi/11;  
topi[-2.87] := -11Pi/12; topi[-2.88] := -11Pi/12;  
topi[-3.14] := -Pi; topi[.26] := Pi/12; topi[.28] := Pi/11;  
topi[.29] := Pi/11; topi[.31] := Pi/10; topi[.34] := Pi/9;  
topi[.35] := Pi/9; topi[.39] := Pi/8; topi[.44] := Pi/7;  
topi[.45] := Pi/7; topi[.52] := Pi/6; topi[.57] := 2Pi/11; topi[.62] :=  
Pi/5; topi[.63] := Pi/5; topi[.69] := 2Pi/9; topi[.70] := 2Pi/9;  
topi[.78] := Pi/4; topi[.79] := Pi/4; topi[.85] := 3Pi/11;  
topi[.86] := 3Pi/11; topi[.94] := 3Pi/10; topi[1.04] := Pi/3;  
topi[1.05] := Pi/3; topi[1.14] := 4Pi/11; topi[1.17] := 3Pi/8;  
topi[1.18] := 3Pi/8; topi[1.25] := 2Pi/5; topi[1.26] := 2Pi/5;  
topi[1.30] := 5Pi/12; topi[1.31] := 5Pi/12; topi[1.34] := 3Pi/7;  
topi[1.35] := 3Pi/7; topi[1.39] := 4Pi/9; topi[1.40] := 4Pi/9;  
topi[1.42] := 5Pi/11; topi[1.43] := 5Pi/11; topi[1.57] := Pi/2;  
topi[1.71] := 6Pi/11; topi[1.74] := 5Pi/9; topi[1.75] := 5Pi/9;  
topi[1.79] := 4Pi/7; topi[1.80] := 4Pi/7; topi[1.83] := 7Pi/12;
```

topi[1.88]:=3Pi/5; topi[1.96]:=5Pi/8; topi[1.99]:=7Pi/11;
topi[2.00]:=7Pi/11; topi[2.09]:=2Pi/3; topi[2.19]:=7Pi/10;
topi[2.20]:=7Pi/10; topi[2.24]:=5Pi/7; topi[2.28]:=8Pi/11;
topi[2.35]:=3Pi/4; topi[2.36]:=3Pi/4; topi[2.44]:=7Pi/9;
topi[2.51]:=4Pi/5; topi[2.57]:=9Pi/11; topi[2.61]:=5Pi/6;
topi[2.62]:=5Pi/6; topi[2.69]:=6Pi/7; topi[2.74]:=7Pi/8;
topi[2.75]:=7Pi/8; topi[2.79]:=8Pi/9; topi[2.82]:=9Pi/10;
topi[2.83]:=9Pi/10; topi[2.85]:=10Pi/11; topi[2.86]:=10Pi/11;
topi[2.87]:=11Pi/12; topi[2.88]:=11Pi/12; topi[3.14]:=Pi;
topi[3.40]:=13Pi/12; topi[3.42]:=12Pi/11; topi[3.43]:=12Pi/11;
topi[3.45]:=11Pi/10; topi[3.46]:=11Pi/10; topi[3.49]:=10Pi/9;
topi[3.59]:=8Pi/7; topi[3.53]:=9Pi/8; topi[3.66]:=7Pi/6;
topi[3.67]:=7Pi/6; topi[3.71]:=13Pi/11; topi[3.76]:=6Pi/5;
topi[3.77]:=6Pi/5; topi[3.83]:=11Pi/9; topi[3.84]:=11Pi/9;
topi[3.92]:=5Pi/4; topi[3.93]:=5Pi/4; topi[3.99]:=14Pi/11;
topi[4.00]:=14Pi/11; topi[4.03]:=9Pi/7; topi[4.04]:=9Pi/7;
topi[4.08]:=13Pi/10; topi[4.18]:=4Pi/3; topi[4.19]:=4Pi/3;
topi[4.28]:=15Pi/11; topi[4.31]:=11Pi/8; topi[4.32]:=11Pi/8;
topi[4.39]:=7Pi/5; topi[4.40]:=7Pi/5; topi[4.45]:=17Pi/12;
topi[4.48]:=10Pi/7; topi[4.49]:=10Pi/7; topi[4.53]:=13Pi/9;
topi[4.54]:=13Pi/9; topi[4.56]:=16Pi/11; topi[4.57]:=16Pi/11;
topi[4.71]:=3Pi/2; topi[4.85]:=17Pi/11; topi[4.86]:=17Pi/11;
topi[4.88]:=14Pi/9; topi[4.89]:=14Pi/9; topi[4.93]:=11Pi/7;
topi[4.94]:=11Pi/7; topi[4.97]:=19Pi/12; topi[5.02]:=8Pi/5;
topi[5.03]:=8Pi/5; topi[5.10]:=13Pi/8; topi[5.11]:=13Pi/8;
topi[5.14]:=18Pi/11; topi[5.23]:=5Pi/3; topi[5.24]:=5Pi/3;
topi[5.34]:=17Pi/10; topi[5.38]:=12Pi/7; topi[5.39]:=12Pi/7;
topi[5.42]:=19Pi/11; topi[5.43]:=19Pi/11; topi[5.49]:=7Pi/4;
topi[5.50]:=7Pi/4; topi[5.58]:=16Pi/9; topi[5.59]:=16Pi/9;
topi[5.65]:=9Pi/5; topi[5.71]:=20Pi/11; topi[5.75]:=11Pi/6;
topi[5.76]:=11Pi/6; topi[5.83]:=13Pi/7; topi[5.89]:=15Pi/8;
topi[5.93]:=17Pi/9; topi[5.96]:=19Pi/10; topi[5.97]:=19Pi/10;
topi[5.99]:=21Pi/11; topi[6.00]:=21Pi/11; topi[6.02]:=23Pi/12;
topi[6.28]:=2Pi; topi[6.54]:=25Pi/12; topi[6.56]:=23Pi/11;
topi[6.57]:=23Pi/11; topi[6.59]:=21Pi/10; topi[6.60]:=21Pi/10;
topi[6.63]:=19Pi/9; topi[6.67]:=17Pi/8; topi[6.68]:=17Pi/8;
topi[6.73]:=15Pi/7; topi[6.80]:=13Pi/6; topi[6.81]:=13Pi/6;
topi[6.85]:=24Pi/11; topi[6.91]:=11Pi/5; topi[6.98]:=20Pi/9;
topi[7.06]:=9Pi/4; topi[7.07]:=9Pi/4; topi[7.13]:=25Pi/11;
topi[7.14]:=25Pi/11; topi[7.18]:=16Pi/7; topi[7.22]:=23Pi/10;
topi[7.23]:=23Pi/10; topi[7.33]:=7Pi/3; topi[7.42]:=26Pi/11;
topi[7.43]:=26Pi/11; topi[7.46]:=19Pi/8; topi[7.53]:=12Pi/5;
topi[7.54]:=12Pi/5; topi[7.59]:=29Pi/12; topi[7.62]:=17Pi/7;
topi[7.63]:=17Pi/7; topi[7.67]:=22Pi/9; topi[7.68]:=22Pi/9;
topi[7.71]:=27Pi/11; topi[7.85]:=5Pi/2; topi[7.99]:=28Pi/11;
topi[8.00]:=28Pi/11; topi[8.02]:=23Pi/9; topi[8.03]:=23Pi/9;
topi[8.07]:=18Pi/7; topi[8.08]:=18Pi/7; topi[8.11]:=31Pi/12;
topi[8.12]:=31Pi/12; topi[8.16]:=13Pi/5; topi[8.17]:=13Pi/5;
topi[8.24]:=21Pi/8; topi[8.25]:=21Pi/8; topi[8.28]:=29Pi/11;
topi[8.37]:=8Pi/3; topi[8.38]:=8Pi/3; topi[8.48]:=27Pi/10;
topi[8.52]:=19Pi/7; topi[8.53]:=19Pi/7; topi[8.56]:=30Pi/11;
topi[8.57]:=30Pi/11; topi[8.63]:=11Pi/4; topi[8.64]:=11Pi/4;

topi[8.72]:=25Pi/9; topi[8.73]:=25Pi/9; topi[8.79]:=14Pi/5;
topi[8.80]:=14Pi/5; topi[8.85]:=31Pi/11; topi[8.90]:=17Pi/6;
topi[8.97]:=20Pi/7; topi[8.98]:=20Pi/7; topi[9.03]:=23Pi/8;
topi[9.07]:=26Pi/9; topi[9.08]:=26Pi/9; topi[9.11]:=29Pi/10;
topi[9.13]:=32Pi/11; topi[9.14]:=32Pi/11; topi[9.16]:=35Pi/12;
topi[9.42]:=3Pi; topi[x_]:=x;

VII. FILE POLAR1.MA

(* Copyright Dennis A. Polaski, May 4, 1993 *)

```
polar2:=Module[ (
),
cltranscend;
func1=Input["Enter one of the polar curves which bounds the
region.

r = "];
func1=func1 /. sqrt->Sqrt;
func1=func1 /. pi->Pi;
func1=func1 /. q->Q;

func2=Input["Enter the other polar curve which bounds the
region.

r = "];
func2=func2 /. sqrt->Sqrt;
func2=func2 /. pi->Pi;
func2=func2 /. q->Q;

limitsknown=Input["If the limits are known type y,
otherwise type n."];
If[limitsknown===y, {
lowerlimit=Input["Input lower limit."];
lowerlimit=lowerlimit /. sqrt->Sqrt;
lowerlimit=lowerlimit /. pi->Pi;
upperlimit=Input["Input upper limit."];
upperlimit=upperlimit /. sqrt->Sqrt;
upperlimit=upperlimit /. pi->Pi;

If[N[upperlimit]<N[lowerlimit],lowerlimit=N[lowerlimit-2Pi]]
},
{lowerlimit=0; upperlimit=2 N[Pi]}}];
bothfunc={func1, func2};
ptranscend;

If[{func1, func2}!=({func1, func2} /. Sqrt[x___]->x),
theroots=NRoots[func1^2==func2^2,Q],
theroots=NRoots[func1==func2, Q];
];

If[theroots!=False, {
theroots=Table[theroots[[i,2]],{i,Length[theroots]}}];
theroots>DeleteCases[theroots, Q_Complex /; Abs[Im[Q]]>.1];
theroots=ReplaceAll[theroots, Q_ -> Re[Q] ];
rmtranscend;
Q=.;
```

```

theroots=Table[FindRoot[func1==func2, {Q,theroots[[m]]}],
{m, Length[theroots]}];
theroots= Q /. theroots;
theroots=N[Union[theroots, {}]];
theroots=Union[theroots, theroots + N[Table[2Pi,
{j, Length[theroots]}]]];
testroots=Union[Select[N[Round[theroots*100]/100],
{#>=N[-Pi] && #<=N[(5/2) Pi]}&],{}];
theroots=Select[theroots, {#<= N[upperlimit] &&
#>= N[lowerlimit]}&];
theroots=Union[N[(Round[100*theroots])/100],{}];
theroots=Map[topi, theroots];
testroots=Map[topi, testroots];
]];

```

```

theroots=Transpose[Select[Table[{theroots[[i]],
(N[Abs[func1-func2] /. Q-> theroots[[i]] )<.05},
{i,1,Length[theroots]}],{#[[2]]===True}&]][[1]];
If[theroots==={}][[1]],theroots={};

```

```

max={0,0};
For[rr=1, rr<=2, {rval=rr; ++rr;
max[[rval]]=Max[DeleteCases[Table[bothfunc[[rval]]
/. Q->values, {values, 0, 2Pi, .25}], x_Complex]]]];
max=Max[max];

```

plotroutine;

```

ptranscend;
For[hh=1, hh<=2, {hval=hh; ++hh;
otherroots=NSolve[{r==0, r==bothfunc[[hval]}], {r,Q}];
otherroots=N[{r,Q} /. otherroots];

```

```

If[otherroots!={r,Q} && otherroots !={},{
otherroots=DeleteCases[otherroots, {r_,Q_Complex /;
Abs[Im[Q]]>.1}];
otherroots=ReplaceAll[otherroots, {r_,Q_} -> {r,Re[Q]}];
rmtranscend;
Q=.;

```

```

otherroots=Table[FindRoot[bothfunc[[hval]]==0,
{Q,otherroots[[m,2]}],{m,Length[otherroots]}];
otherroots={0,Re[Q]} /. otherroots;
otherroots=N[Union[otherroots, {}]];
otherroots=Union[otherroots, otherroots+ N[Table[{0,2Pi},
{i,Length[otherroots]}]]];
otherroots=Select[otherroots, {#[[2]]<=N[upperlimit] &&
#[[2]] >= N[lowerlimit]}&];
otherroots=Union[N[(Round[100*otherroots])/100],{}];
]];

```

```

otherroots=Union[ReplaceAll[otherroots,{0,6.28}->{0,0}],{}];
otherroots=Union[ReplaceAll[otherroots,{Q,r}->{{Q,r}}],{}];
If[hval==1,func1roots=otherroots,func2roots=otherroots];
]];

```

```

allroots={func1roots,func2roots};
endprogram=0;
runoption=0;
diffcheck;

```

```

If[(theroots===False || testdiff===yes) && runoption==0 &&
Length[func1roots]<=1 && Length[func2roots]<=1,{
If[limitsknown===n,{
If[Length[theroots]==1,
Print["
                                ",
"The two polar curves intersect at Q = ",
topi[theroots[[1]] ] ]];
If[Length[theroots] >1,
Print["The two polar curves intersect at Q:           ","
",
Map[topi,theroots] ]];
Pause[8];
newcurve=Input["The polar curves are plotted from Q=0
to Q=2Pi. If you want to change the limits type y, otherwise
type n."];

```

```

If[newcurve===y,{
lowerlimit=Input["Input new lower limit."];
lowerlimit=lowerlimit /. sqrt->Sqrt;
lowerlimit=lowerlimit /. pi->Pi;
While[lowerlimit <-N[Pi],{lowerlimit=Input["The lower limit
can not be less than -Pi, input lower limit again."];
lowerlimit=lowerlimit /. sqrt->Sqrt;
lowerlimit=lowerlimit /. pi->Pi;
}];
upperlimit=Input["Input new upper limit."];
upperlimit=upperlimit /. sqrt->Sqrt;
upperlimit=upperlimit /. pi->Pi;
While[upperlimit>2 N[Pi],{upperlimit=Input["The upperlimit
can not be greater than 2Pi, input upper limit again."];
upperlimit=upperlimit /. sqrt->Sqrt;
upperlimit=upperlimit /. pi->Pi;
}];
If[N[upperlimit]<N[lowerlimit],lowerlimit=N[lowerlimit-2Pi]];
rmtranscend; plotroutine,
}];
}];
complexcheck; negcheck;
If[Length[complexfunc1]<=2 && Length[complexfunc2]<=2 &&
endprogram==0, findarea2];
If[(Length[complexfunc1] >2 || Length[complexfunc2] >2) &&

```

```

endprogram==0, findareal]; runoption=1
});

If[testdiff===yes && (Length[func1roots] >1 ||
Length[func2roots]>1) && runoption==0, (
If[limitsknown===n, (
If[Length[theroots]==1,
Print["
                                ",
"The two polar curves intersect at Q = ",
topi[theroots[[1]] ] ]];
If[Length[theroots] >1,
Print["The two polar curves intersect at Q:          ","
                                ",
                                " Map[topi, theroots] ]];
Pause[8];
newcurve=Input["The polar curves are plotted from Q=0
to Q=2Pi. If you want to change the limits type y, otherwise
type n."];

If[newcurve===y, (
lowerlimit=Input["Input new lower limit."];
lowerlimit=lowerlimit /. sqrt->Sqrt;
lowerlimit=lowerlimit /. pi->Pi;
While[lowerlimit <-N[Pi], {lowerlimit=Input["The lower limit
can not be less than -Pi, input lower limit again."];
lowerlimit=lowerlimit /. sqrt->Sqrt;
lowerlimit=lowerlimit /. pi->Pi;
}];
upperlimit=Input["Input new upper limit."];
upperlimit=upperlimit /. sqrt->Sqrt;
upperlimit=upperlimit /. pi->Pi;
While[upperlimit>2 N[Pi], {upperlimit=Input["The upperlimit
can not be greater than 2Pi, input upper limit again."];
upperlimit=upperlimit /. sqrt->Sqrt;
upperlimit=upperlimit /. pi->Pi;
}];
If[N[upperlimit]<N[lowerlimit],lowerlimit=N[lowerlimit-2Pi]];
rmtranscend; plotroutine;
});
});
complexcheck; negcheck;
If[Length[complexfunc1]<=2 && Length[complexfunc2]<=2 &&
endprogram==0, findarea2];
If[(Length[complexfunc1] >2 || Length[complexfunc2] >2) &&
endprogram==0, findareal]; runoption=1
});

If[(theroots===False || Length[theroots] <=1) &&
runoption==0 &&
(Length[func1roots] >1 || Length[func2roots]>1), (
If[limitsknown===n, (

```

```

If[Length[theroots]==1,
Print["
", "
The two polar curves intersect at Q = ",
topi[theroots[[1]] ] ]];
Pause[8];
newcurve=Input["The polar curves are plotted from Q=0
to Q=2Pi. If you want to change the limits type y, otherwise
type n."];
If[newcurve===y,{
lowerlimit=Input["Input new lower limit."];
lowerlimit=lowerlimit /. sqrt->Sqrt;
lowerlimit=lowerlimit /. pi->Pi;
While[lowerlimit <-N[Pi], {lowerlimit=Input["The lower limit
can not be less than -Pi, input lower limit again."];
lowerlimit=lowerlimit /. sqrt->Sqrt;
lowerlimit=lowerlimit /. pi->Pi;
}];
upperlimit=Input["Input new upper limit."];
upperlimit=upperlimit /. sqrt->Sqrt;
upperlimit=upperlimit /. pi->Pi;
While[upperlimit>2 N[Pi], {upperlimit=Input["The upperlimit
can not be greater than 2Pi, input upper limit again."];
upperlimit=upperlimit /. sqrt->Sqrt;
upperlimit=upperlimit /. pi->Pi;
}];
If[N[upperlimit]<N[lowerlimit],lowerlimit=N[lowerlimit-2Pi]];
rmtranscend; plotroutine;
}];
}];
}];
complexcheck; negcheck;
If[Length[complexfunc1]<=2 && Length[complexfunc2]<=2 &&
endprogram==0, findarea2];
If[(Length[complexfunc1] >2 || Length[complexfunc2] >2) &&
endprogram==0, findarea1];runoption=1
}];

control=0;

If[Length[theroots]>=2 && limitsknown===n && runoption==0,{
Print["The two polar curves intersect at Q:
", "
", Map[topi, theroots] ];
Pause[8];
newcurve=Input["The polar curves are plotted from Q=0
to Q=2Pi. If you want to change the limits type y, otherwise
type n."];
If[newcurve===y,{
lowerlimit=Input["Input new lower limit."];
lowerlimit=lowerlimit /. sqrt->Sqrt;
lowerlimit=lowerlimit /. pi->Pi;
While[lowerlimit <-N[Pi], {lowerlimit=Input["The lower limit
can not be less than -Pi, input lower limit again."];

```



```

lowerlimit=lowerlimit /. sqrt->Sqrt;
lowerlimit=lowerlimit /. pi->Pi;
});
upperlimit=Input["Input new upper limit."];
upperlimit=upperlimit /. sqrt->Sqrt;
upperlimit=upperlimit /. pi->Pi;
While[upperlimit>2 N[Pi], {upperlimit=Input["The upperlimit
can not be greater than 2Pi, input upper limit again."];
upperlimit=upperlimit /. sqrt->Sqrt;
upperlimit=upperlimit /. pi->Pi;
});
If[N[upperlimit]<N[lowerlimit],lowerlimit=N[lowerlimit-2Pi]];
});
badlimits=Select[N[theroots],
(#>N[lowerlimit] && #<N[upperlimit])&];
While[Length[badlimits]>0, {
If[control==1, {
Print["For Q=-Pi to Q=5/2 Pi the two polar curves intersect
at Q:", "", Map[topi, testroots] ]; Pause[8]};
control=control+1;
lowerlimit=Input["A point of intersection can not lie
between the two limits. Input new lower limit."];
lowerlimit=lowerlimit /. sqrt->Sqrt;
lowerlimit=lowerlimit /. pi->Pi;
upperlimit=Input["Input new upper limit."];
upperlimit=upperlimit /. sqrt->Sqrt;
upperlimit=upperlimit /. pi->Pi;
If[N[upperlimit]<N[lowerlimit],lowerlimit=N[lowerlimit-2Pi]];
badlimits=Select[N[testroots],
(#>N[lowerlimit] && #<N[upperlimit])&];
});
rmtranscend; plotroutine; complexcheck; negcheck;
If[(Length[complexfunc1]<=2 && Length[complexfunc2]<=2) &&
endprogram==0,findarea2];
If[(Length[complexfunc1] >2 || Length[complexfunc2] >2) &&
endprogram==0,findarea1]; runoption=1
});

onceprint=0;
If[Length[theroots]>=1 && limitsknown===y && runoption==0,{
badlimits=Select[N[theroots],
(#>N[lowerlimit] && #<N[upperlimit])&];

While[Length[badlimits]>0, {
If[onceprint==0,{
Print["The two polar curves intersect at Q:           ","
", Map[topi,theroots] ]; onceprint=1}];
Pause[8];
If[control==1, {
Print["For Q=-Pi to Q=5/2 Pi the two polar curves intersect
at Q:", "", Map[topi, testroots] ]; Pause[8]};

```

```

control=control+1;
lowerlimit=Input["A point of intersection can not lie
between the two limits. Input new lower limit."];
lowerlimit=lowerlimit /. sqrt->Sqrt;
lowerlimit=lowerlimit /. pi->Pi;
upperlimit=Input["Input new upper limit."];
upperlimit=upperlimit /. sqrt->Sqrt;
upperlimit=upperlimit /. pi->Pi;
If[N[upperlimit]<N[lowerlimit],lowerlimit=N[lowerlimit-2Pi]];
badlimits=Select[N[testroots],
(#>N[lowerlimit] && #<N[upperlimit])&];
If[Length[badlimits]==0, {
rmtranscend; plotroutine});
}];
complexcheck; negcheck;
If[(Length[complexfunc1]<=2 && Length[complexfunc2]<=2) &&
endprogram==0,findarea2];
If[(Length[complexfunc1] >2 || Length[complexfunc2] >2) &&
endprogram==0,findarea1]; runoption=1
});
];

```

```

complexcheck:=Module[{}],
complexfunc1=DeleteCases[DeleteCases[Table[N[func1 /.
Q->values], {values, N[lowerlimit], N[upperlimit], .1}],
r_Integer],r_Real];
complexfunc2=DeleteCases[DeleteCases[Table[N[func2 /.
Q->values], {values,N[lowerlimit],N[upperlimit], .1}],
r_Integer], r_Real] ];

```

```

diffcheck:=Module[{}],
difference=DeleteCases[Table[func1-func2 /.Q->values,
{values, lowerlimit+.01, upperlimit-.01,.25}], x_Complex];
If[Max[difference]<=0 && Min[difference]<0 ||
Max[difference]>0 && Min[difference]>=0,
testdiff=yes, testdiff=no] ];

```

```

findareal:=Module[{}],
For[tt=1, tt<=2, {tval=tt; ++tt;
If[allroots[[tval]] != {{Q,r}}, {
otherroots=Transpose[ allroots[[tval]] ]];
otherroots=otherroots[[2]];
otherroots=Sort[Union[ReplaceAll[Join[{N[lowerlimit]},
otherroots, {N[upperlimit]}], 0.->0],{}}];
otherroots=Select[otherroots, (#>=N[lowerlimit] &&
#<=N[upperlimit])&];
},{otherroots=ReplaceAll[{N[lowerlimit],N[upperlimit]},
0.->0]}}];
rmtranscend; area=.; totalarea={};
For[vv=1, vv<=Length[otherroots]-1, {vval=vv; ++vv;
Q=(otherroots[[vval]] + otherroots[[vval +1]])/2;
If[ N[bothfunc[[tval]] ]>0, {Q=.;

```

```

area= NIntegrate[.5* bothfunc[[tval]]^2, {Q,
topi[otherroots[[vval]] ], topi[otherroots[[vval+1]] ]}];
AppendTo[totalarea, Abs[area] ]];
]];
area= Apply[Plus, totalarea];
If[tval==1, areal=area];
If[tval==2, area2=area];
]];
area= area2 - areal; cltranscend;Q=.;
printfunc1 ];

findarea2:=Module[{},
Q=.; rmtranscend; area=NIntegrate[.5*(func2^2 - func1^2),
{Q, topi[lowerlimit], topi[upperlimit]}]; printfunc2];

printfunc1:=Module[{},
If[area<=0, {temp=func1; func1=func2; func2=temp}];
Print[" "];
Print[" "];
Print["The outer polar curve is r = ", func2];
Print["The inner polar curve is r = ", func1];
Print["The area of the region bounded by the two"];
Print["polar curves is ",Abs[area]] ];

printfunc2:=Module[{},
cltranscend;
If[area<=0, {temp=func1; func1=func2; func2=temp}];
Print[" "];
Print[" "];
Print["The outer polar curve is r = ", func2];
Print["The inner polar curve is r = ", func1];
Print["The limits of integration are a=",
topi[N[Round[lowerlimit*100]/100]],
" and b=", topi[N[Round[upperlimit*100]/100]] ];
Print["The area of the region bounded by the two"];
Print["polar curves is ", Abs[area]] ];

negcheck:= Module[{},
negfunc1=Min[DeleteCases[Table[func1 /. Q->values,
{values, lowerlimit+.01, upperlimit-.01, .25}], x_Complex]];
negfunc2=Min[DeleteCases[Table[func2 /. Q->values,
{values, lowerlimit+.01, upperlimit-.01, .25}], x_Complex]];
If[negfunc1<0 || negfunc2<0,{cltranscend;
If[negfunc1<0 && negfunc2<0,
{Print["Both functions have negative values for r."];
Print["This program can not solve such problems."]},
If[negfunc1<0,
{Print["The function r = ",func1," has negative values."];
Print["This program can not solve such problems." ]},
{Print["The function r = ",func2," has negative values."];
Print["This program can not solve such problems."}}}
];

```

```

endprogram=1
]]
];

```

```

plotroutine:=

```

```

Module[(),
plotchunk=PolarPlot[{func1, func2}, {Q, N[lowerlimit],
N[upperlimit]}, DisplayFunction->Identity, AxesLabel->
{"x", "y"}, AspectRatio->Automatic, PlotRange->
{{-1.02 max, 1.02 max}, {-1.02 max, 1.02 max}}];

```

```

ray1=Re[N[(func1 /. Q-> lowerlimit)]];
ray2=Re[N[(func1 /. Q-> upperlimit)]];
ray3=Re[N[(func2 /. Q-> lowerlimit)]];
ray4=Re[N[(func2 /. Q-> upperlimit)]];
llequivalent=N[Mod[lowerlimit, 2Pi]];
ulequivalent=N[Mod[upperlimit, 2Pi]];

```

```

If[lowerlimit < 0, llequivalent= 2 N[Pi]+lowerlimit,
llequivalent=lowerlimit];
If[upperlimit < 0, ulequivalent= 2 N[Pi]+upperlimit,
ulequivalent=upperlimit];

```

```

Q=.;

```

```

If[(ray1==ray2 && llequivalent==ulequivalent &&
ray3==ray4) || (ray1==0 && ray2==0 && ray3==0 && ray4==0),
{Show[plotchunk, DisplayFunction-> $DisplayFunction]},
{Show[plotchunk, Graphics[{
Line[{{Cos[lowerlimit]*ray1, Sin[lowerlimit]*ray1},
{Cos[lowerlimit]*ray3, Sin[lowerlimit]*ray3}}],
Line[{{Cos[upperlimit]*ray2, Sin[upperlimit]*ray2},
{Cos[upperlimit]*ray4, Sin[upperlimit]*ray4}}],
}], DisplayFunction-> $DisplayFunction]
}]]

```

```

ptranscend:=Module[(),

```

```

sin[x_]:= x - x^3/6 + x^5/120 - x^7/5040 + x^9/362880 -
x^11/39916800+x^13/6227020800-x^15/1307674368000+
x^17/355687428096000-x^19/121645100408832000;
cos[x_]:= 1 - x^2/2 + x^4/24 - x^6/720 + x^8/40320 -
x^10/3628800 + x^12/479001600;
tan[x_]:= x + x^3/3 + 2x^5/15 + 17x^7/315 + 62x^9/2835 +
1382x^11/155925;
cot[x_]:= 1/x -x/3 -x^3/45 - 2x^5/945 - x^7/4725 -
2x^9/93555;
csc[x_]:= 1/x + x/6 + 7x^3/360 + 31x^5/15120 +
127x^7/604800 + 73x^9/3421440;
sec[x_]:= 1 + x^2/2 + 5x^4/24 + 61x^6/720 + 277x^8/8064 +
50521x^10/3628800;

```

```
exp[x_] := 1 + x + x^2/2 + x^3/6 + x^4/24 + x^5/120 + x^6/720 +  
x^7/5040 + x^8/40320 + x^9/362880]
```

VIII. FILE POLPLOT1.MA

(* Copyright Dennis A. Polaski, May 4,1993 *)

```

polplot1:=
Module[{
    func, limitsknown, theroots, lowerlimit,
    upperlimit, newcurve, curvesknown, temp
},
Q=.;
cltranscend;
func=Input["Enter the polar curve which bounds the region.

r = "];
func=func /. sqrt->Sqrt;
func=func /. pi->Pi;
func=func /. q->Q;
revfunc=Replace[func, Sqrt[x_] -> x];
limitsknown=Input["If the limits are known type y,
otherwise type n."];

If[limitsknown===y,
{lowerlimit=Input["Input lower limit."];
lowerlimit=lowerlimit /. sqrt->Sqrt;
lowerlimit=lowerlimit /. pi->Pi;
While[lowerlimit <-N[Pi], {lowerlimit=Input["The lowerlimit
can not be less than -Pi, input lower limit again."];
lowerlimit=lowerlimit /. sqrt->Sqrt;
lowerlimit=lowerlimit /. pi->Pi;
}}];
upperlimit=Input["Input upper limit."];
upperlimit=upperlimit /. sqrt->Sqrt;
upperlimit=upperlimit /. pi->Pi;
While[upperlimit>2 N[Pi], {upperlimit=Input["The upperlimit
can not be greater than 2Pi, input upper limit again."];
upperlimit=upperlimit /. sqrt->Sqrt;
upperlimit=upperlimit /. pi->Pi;
}}];
},
{lowerlimit=0.; upperlimit= 2Pi}];

rmtranscend;
max=Max[DeleteCases[Table[func /. Q -> values,
{values, 0, 2Pi, .25}],x_Complex]];

ray1=N[(func /. Q->lowerlimit)];
ray2=N[(func /. Q->upperlimit)];

```

```

l1equivalent=N[Mod[lowerlimit, 2Pi]];
ulequivalent=N[Mod[upperlimit, 2Pi]];

If[lowerlimit < 0, l1equivalent= 2 N[Pi]+lowerlimit,
l1equivalent= lowerlimit];
If[upperlimit < 0, ulequivalent= 2 N[Pi]+upperlimit,
ulequivalent= upperlimit];

Q=.;
If[(ray1==ray2) && (l1equivalent==ulequivalent)||
(ray1==0 && ray2==0),{
PolarPlot[func,{Q, lowerlimit, upperlimit},
AxesLabel -> {"x","y"}, AspectRatio -> Automatic,
PlotRange-> {{-1.02 max, 1.02 max},{-1.02 max, 1.02 max}}],
{
Show[PolarPlot[func,{Q, lowerlimit, upperlimit},
DisplayFunction -> Identity, AxesLabel -> {"x","y"},
AspectRatio -> Automatic, PlotRange-> {{-1.02 max,1.02 max},
{-1.02 max, 1.02 max}}],
Graphics[{
Line[{{0,0},{Cos[lowerlimit]*ray1, Sin[lowerlimit]*ray1}],
Line[{{0,0},{Cos[upperlimit]*ray2, Sin[upperlimit]*ray2}],

}],
DisplayFunction-> $DisplayFunction}}];

ptranscend;
theroots=NSolve[{r==0, r==revfunc},{r,Q}];
theroots=N[{r,Q} /. theroots];

If[theroots!={r,Q} && theroots !={},{
theroots>DeleteCases[theroots,{r_,Q_Complex /;
Abs[Im[Q]]>.1}];
theroots=ReplaceAll[theroots,{r_,Q_} -> {r,Re[Q]}];
rmtranscend;
Q=.;
theroots=Table[FindRoot[revfunc==0,
{Q,theroots[[m,2]}]],{m,Length[theroots]}];
theroots={0,Q} /. theroots;
theroots=N[Union[theroots,{}]];
theroots=Union[theroots, theroots+ N[Table[{0,2Pi},
{i,Length[theroots]}]]];
theroots=Select[theroots,#[[2]]<=N[upperlimit] &&
#[[2]] >= N[lowerlimit]]&;
theroots=Union[N[(Round[100*theroots])/100],{}];
}];
rmtranscend;

If[limitsknown!=y, {
If[theroots!={r,Q} && theroots!={} &&
Length[theroots]>=1,{
Print[" "];

```

```

Print["The polar curve r is equal to zero when Q is:
      ",Map[topi,Transpose[theroots][[2]] ] ]
]];

Pause[4];
newcurve=Input["The polar curve is plotted from Q=0
to Q=2 Pi. If you want to change the limits type y, otherwise
type n."];

If[newcurve===y,{
lowerlimit=Input["Input new lower limit."];
lowerlimit=lowerlimit /. sqrt->Sqrt;
lowerlimit=lowerlimit /. pi->Pi;
While[lowerlimit <-N[Pi], {lowerlimit=Input["The lower limit
can not be less than -Pi, input lower limit again."];
lowerlimit=lowerlimit /. sqrt->Sqrt;
lowerlimit=lowerlimit /. pi->Pi;
}];
upperlimit=Input["Input new upper limit."];
upperlimit=upperlimit /. sqrt->Sqrt;
upperlimit=upperlimit /. pi->Pi;
While[upperlimit>2 N[Pi], {upperlimit=Input["The upperlimit
can not be greater than 2Pi, input upper limit again."];
upperlimit=upperlimit /. sqrt->Sqrt;
upperlimit=upperlimit /. pi->Pi;
}];

ray1=N[(func /. Q->lowerlimit)];
ray2=N[(func /. Q->upperlimit)];

llequivalent=N[Mod[lowerlimit, 2Pi]];
ulequivalent=N[Mod[upperlimit, 2Pi]];

If[lowerlimit < 0, llequivalent= 2 N[Pi]+lowerlimit,
llequivalent= lowerlimit];
If[upperlimit < 0, ulequivalent= 2 N[Pi]+upperlimit,
ulequivalent= upperlimit];

Q=.;
If[(ray1==ray2) && (llequivalent==ulequivalent)||
(ray1==0 && ray2==0),{
PolarPlot[func,{Q, lowerlimit, upperlimit},
AxesLabel -> {"x","y"}, AspectRatio -> Automatic,
PlotRange-> {{-1.02 max, 1.02 max},{-1.02 max, 1.02 max}}],
{
Show[PolarPlot[func,{Q, lowerlimit, upperlimit},
DisplayFunction -> Identity, AxesLabel -> {"x","y"},
AspectRatio -> Automatic, PlotRange-> {{-1.02 max,1.02 max},
{-1.02 max, 1.02 max}}],
Graphics[{
Line[{{0,0},{Cos[lowerlimit]*ray1, Sin[lowerlimit]*ray1}]],

```



```
Line[{{0,0},{Cos[upperlimit]*ray2, Sin[upperlimit]*ray2}},  
  ],  
DisplayFunction-> $DisplayFunction]]  
];  
];  
]
```

IX. FILE POLPLOT2.MA

(* Copyright Dennis A. Polaski, May 4, 1993 *)

```

polplot2:=Module[{ },
cltranscend;
func1=Input["Enter one of the polar curves which bounds the
region.

r = "];
func1=func1 /. sqrt->Sqrt;
func1=func1 /. pi->Pi;
func1=func1 /. q->Q;
func2=Input["Enter the other polar curve which bounds the
region.

r = "];
func2=func2 /. sqrt->Sqrt;
func2=func2 /. pi->Pi;
func2=func2 /. q->Q;

limitsknown=Input["If the limits are known type y,
otherwise type n."];
If[limitsknown===y, {
lowerlimit=Input["Input lower limit."];
lowerlimit=lowerlimit /. sqrt->Sqrt;
lowerlimit=lowerlimit /. pi->Pi;
upperlimit=Input["Input upper limit."];
upperlimit=upperlimit /. sqrt->Sqrt;
upperlimit=upperlimit /. pi->Pi},
{lowerlimit=0; upperlimit=2 N[Pi]}}];
bothfunc={func1, func2};
ptrascend;

If[{{func1, func2}!={{func1, func2} /. Sqrt[x___]->x),
theroots=NRoots[func1^2==func2^2, Q],
theroots=NRoots[func1==func2, Q];
];

If[theroots!=False, {
theroots= Table[theroots[[i,2]],{i,Length[theroots]}}];
theroots=DeleteCases[theroots, Q_Complex /; Abs[Im[Q]]>.1];
theroots=ReplaceAll[theroots, Q_ -> Re[Q] ];
rmtranscend;
Q=.;
theroots=Table[FindRoot[func1==func2, {Q,theroots[[m]]} ],
{m, Length[theroots]}}];
theroots= Q /. theroots;
theroots=N[Union[theroots, {}]];

theroots=Union[theroots, theroots + N[Table[2Pi,

```

```

{j, Length[theroots]]}];
testroots=Union[Select[N[Round[theroots*100]/100],
(#>=N[-Pi] && #<=N[(5/2) Pi])&],{}];
theroots=Select[theroots, (#<= N[upperlimit] &&
#>= N[lowerlimit])&];
theroots=Union[N[(Round[100*theroots])/100],{}];
)];

theroots=Transpose[Select[Table[{theroots[[i]],
(Abs[func1-func2] /. Q-> theroots[[i]])<.01},
{i,1,Length[theroots]}],(#[[2]]===True)&]][[1]];
If[theroots==={}[[1]],theroots={});

max={0,0};
For[rr=1, rr<=2, {rval=rr; ++rr;
max[[rval]]=Max[DeleteCases[Table[bothfunc[[rval]]
/. Q->values, {values, 0, 2Pi, .25}], x_Complex] ]];
max=Max[max];
plotroutine;

If[limitsknown===n, {
If[Length[theroots]==1,
Print["", "
The two polar curves intersect at Q = ",
topi[theroots[[1]]] ]];
If[Length[theroots] >1,
Print["The two polar curves intersect at Q: ", "
", Map[topi,theroots] ]];
Pause[1];Pause[4];
newcurve=Input["The polar curves are plotted from Q=0
to Q=2Pi. If you want to change the limits type y, otherwise
type n."];
If[newcurve===y,{
lowerlimit=Input["Input new lower limit."];
lowerlimit=lowerlimit /. sqrt->Sqrt;
lowerlimit=lowerlimit /. pi->Pi;
While[lowerlimit <-N[Pi], {lowerlimit=Input["The lower limit
can not be less than -Pi, input lower limit again."];
lowerlimit=lowerlimit /. sqrt->Sqrt;
lowerlimit=lowerlimit /. pi->Pi;
}];
upperlimit=Input["Input new upper limit."];
upperlimit=upperlimit /. sqrt->Sqrt;
upperlimit=upperlimit /. pi->Pi;
While[upperlimit>2 N[Pi], {upperlimit=Input["The upperlimit
can not be greater than 2Pi, input upper limit again."];
upperlimit=upperlimit /. sqrt->Sqrt;
upperlimit=upperlimit /. pi->Pi;
}];
rmtranscend; plotroutine
}];
}];
];

```

LIST OF REFERENCES

1. Finney, R. and Thomas, G., Calculus, Addison-Wesley Publishing Company, Reading, Massachusetts, September 1990.
2. Bittinger, M. and Morrel, B., Applied Calculus, Addison-Wesley Publishing Company, Reading, Massachusetts, 1988.
3. Berkey, Dennis, Calculus, Second Edition, Saunders College Publishing, New York, NY, 1988.
4. Blachman, Nancy, Mathematica: A Practical Approach, Prentice-Hall, Inc., Englewood, New Jersey, 1992.
5. Blachman, Nancy, The Mathematica Quick Reference Guide, Variable Symbols, Inc., Berkeley, California, 1990.
6. Wolfram, Stephen, Mathematica: A System for Doing Mathematics by Computer, Addison-Wesley Publishing Company, Reading, Massachusetts, 1991.

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center 2
Cameron Station
Alexandria, VA 22304-6145
2. Library, Code 52 2
Naval Postgraduate School
Monterey, CA 93943-5002
3. Professor David Canright (Code MA/Ca) 2
Naval Postgraduate School
Monterey, CA 93943-5000
4. Professor Richard Franke, (Code MA/Fe) 1
Naval Postgraduate School
Monterey, CA 93943-5000
5. Professor Beny Neta (Code MA/Nd) 2
Naval Postgraduate School
Monterey, CA 93943-5000
6. LTC John Robinson 1
Department of Mathematical Sciences
United States Military Academy
West Point, NY 10996-1786
7. Amy Young 1
Wolfram Research, Inc.
100 Trade Center Drive
Champaign, IL 61820-7237
8. CPT Dennis A. Polaski 5
Department of Mathematical Sciences
United States Military Academy
West Point, NY 10996-1786