

Fully Calculating Autosomal DNA Coverage Recursively

By Wesley Johnston, May 2023

Abstract

In my first paper (“Calculating Autosomal DNA Match Coverage: A generalized additive recursive method”), I gave the pseudo-code for a generalized additive recursive algorithm to accurately calculate the lower bound of the range of DNA coverage for a target person.ⁱ In that paper, Appendix 4 (“Calculating the Shared DNA Component: Dealing with Ranges”) provided an extended set of scenarios showing how the coverage of a person for whom none of their children tested can only accurately be reported as a range, with a lower bound and an upper bound. Like the cone of uncertainty in a hurricane forecast, there is no one single value that can be considered to be the coverage estimate when no children of the target person have tested.ⁱⁱⁱ

Dr. David Stumpf made the first implementation of the algorithm in his Graphs for Genealogists (GFG) software. Jonny Perl then created the DNA Painter “Coverage Estimator” tool, which implemented the method of Paul Woodbury as used by Leah Perle Larkin, a method that propagates the average (instead of the lower bound) up the generations from the test takers to the target ancestor. So, the DNA Painter implementation and both the algorithm in my prior paper and in GFG will give different single numbers, neither of which fully represents the range that is the reality of the coverage for the referenced ancestor in no-child-tested scenarios.^{iv}

In this paper, I extend the algorithm to accurately calculate both the lower and the upper bounds of the range. I also explore the behavior of the ranges and their average and of the propagated average (as used in DNA Painter). I then provide an intuitive understanding of DNA coverage ranges to understand who the best target tester is to improve the overall coverage of the target ancestor.

Review of the Recursive Algorithm

My lower-bound algorithm consisted of two parallel operations. One operation traced all descendants of the target person to see which ones had DNA-tested. Thus, this operation is essentially a top-down tracing. The second operation does a bottom-up calculation of the DNA coverage.^v

1. Count the number of children (N) of the reference person (in the first call, this is the target ancestor).
2. Calculate the number of pieces P of the Venn diagram of combinations of the children (see Figure 1) and the maximum coverage percentage M for each. $P=(2^N)-1$, and $M=1/(2^N)$.
3. Generate the module-internal tables, setting values to be calculated initially at zero.
4. For each child, determine the coverage of the child by calling this same module.
5. For each piece/row p of module-internal table, calculate W(p) and R(p) for that piece/row.
6. Add the percentages for the pieces together (sum the R(p)) to calculate the coverage percentage of the reference person and return that number and exit the module.

The key vehicle for all the calculations is the Venn diagram of all permutations of combinations of the children's DNA contribution to the parent's DNA coverage. The Venn diagram is what enables the recursive calls that can accurately calculate the lower and upper bounds.^{vi}

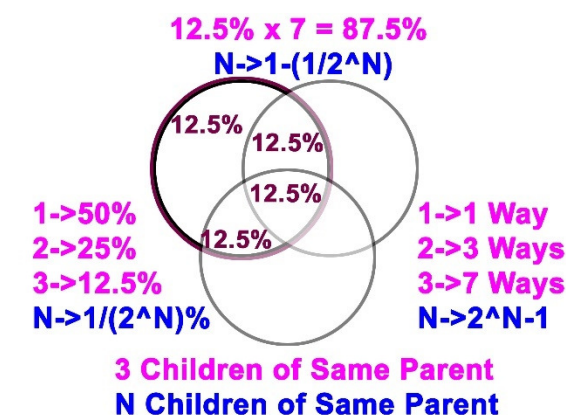


Figure 1 Pieces of DNA Covered by N Children (P = Ways with M% in each Piece)

The module-internal table represents the Venn diagram for the person. For a person with 2 children (thus a Venn diagram of two intersecting circles with one piece unique to each child and the third intersecting piece shared by both children), the table looks like this, where N is the number of children and P is the number of pieces of the Venn diagram so that $P = (2^N) - 1$.

Lower Bound Module-Internal Table - People and Pieces
With Completed W and R Values in Call Level 1

P (Piece)/N (child)	1	2	M	W(p)	R(p)
1	0	1	25%	50%	12.5%
2	1	0	25%	100%	25%
3	1	1	25%	100%	25%

Each row represents a piece of the Venn diagram.

There is a column for each child of the target person and the columns for M, W(p) and R(p). The child column bit value represents the presence (1) or absence (0) of that child in that piece of the Venn diagram. So, row 1 in the example table represents the piece of the Venn diagram that considers only the DNA contribution of child 2.

In this way, each piece of the Venn diagram represents a unique combination of the children: the table is functionally identical to the Venn diagram.

Differences in Calculation of the Lower and Upper Bounds

In this table, child 2's child has tested, and child 1 has tested. So, child 2 has 50% coverage, and child 1 has 100% coverage. The example shows the calculation of the lower bound for the parent of the two children. The lower and upper bounds each require their own Venn diagram and thus their own table to

accurately calculate the bounds of the range of DNA coverage of the target person from the DNA of their children. Column M is the same in both tables. But W and R are calculated differently.

Lower Bound (Max/Product)

In the lower bound case, in pieces of the Venn diagram for combinations of the children's DNA, the worst-case scenario (the lower bound) is the contribution of the child with the largest DNA coverage. This piece of the Venn diagram reflects DNA inherited from the parent by all the children in that piece. So, the parent's own coverage from that piece will be at least the amount due to the contribution of the child who has the most coverage.

Thus, for each row/piece, W (the combined weight of the children's contributions) is calculated simply by taking the coverage from the child with the maximum DNA of any child represented in that row/piece.

$M = 1 / (2^N)$ expressed as a percent (the maximum coverage possible for a piece of the Venn diagram)

$W(p) = \text{Maximum of all products of each child's presence/absence bit times their own coverage, for row } p$

$R(p) = M * W(p)$ for row p

Each weight, $W(p)$, is calculated for its row by using the combination of children specified in the binary columns for that row and multiplying that 0 (absence) or 1 (presence) value by the DNA coverage of that child. The maximum value of each of these products then becomes the value of $W(p)$ for that row p.

For example, in the table's row/piece 1, cell(1,1) is 0, and cell(1,2) is 1. So, $W(1)$ is the greater of (0 * 100%) and (1 * 50%), which is the greater of 0% and 50%. Thus $W(1) = 50\%$.

The Result (R) for that row is the product of the maximum (M) and the amount of coverage of the child with the highest contribution ($W(p)$).

Upper Bound (Sum/Min)

In the upper bound case, in pieces of the Venn diagram for combinations of the children's DNA, the best-case scenario (the upper bound) is the aggregate contribution of all the child. While this piece of the Venn diagram reflects DNA inherited from the parent by all the children in that piece, if they have less than 100% themselves, then the upper bound assumes that the DNA that they inherited in this piece does not match the DNA the other children inherited in this piece. So, the parent's own coverage from this piece will be either (a) the sum of the M-weighted contributions of all the children or (b) the maximum amount (M) that this piece can cover of the parent's DNA, whichever is least.

$M = 1 / (2^N)$ expressed as a percent (the maximum coverage possible for a piece of the Venn diagram)

$W(p) = \text{Sum of all products of each child's presence/absence bit times their own coverage times } M, \text{ for row } p$

$R(p) = \text{Minimum of } M \text{ and } W(p) \text{ for row } p$

Using the same two children as in the lower bound table, the upper bound table looks like this.

**Upper Bound Module-Internal Table - People and Pieces
With Completed W and R Values in Call Level 1**

P (Piece)/N (child)	1	2	M	W(p)	R(p)
1	0	1	25%	12.5%	12.5%
2	1	0	25%	25%	25%
3	1	1	25%	37.5%	25%

For example, in the table's row/piece 3, cell(3,1) is 1, and cell(3,2) is 1. So, W(1) is the sum of (1 * 100% * 25%) and (1 * 50% * 25%) = 25% + 12.5% = 37.5%.

The Result (R) for that row is the minimum of (a) the maximum (M) and (b) the sum (W(p)) = Min(25%, 37.5%) = 25%.

In this case, because one child has DNA-tested, the lower bound and upper bound are the same. In cases where two or more children of a parent have not DNA-tested but do have tested descendants, a single number cannot fully represent the DNA coverage. Only a range, defined by a lower and an upper bound, can fully represent the DNA coverage. In such cases, the lower bound will always be less than the upper bound.

Understanding How the Calculation of Coverage Works in the Recursive Algorithm

The first call of the algorithm considers the target person and his/her children. Each child is then checked to determine their own DNA coverage. If the child has DNA-tested, then their coverage is 100%. Otherwise, the exact same algorithm is called for the child and their children. And this process repeats until all DNA-tested descendants of the target person have been found and their contributions included in the calculation. Their lower and upper bound contributions propagate up through the call levels to allow calculation of the target person's DNA coverage estimate.

Considerations of Using DNA Coverage Estimates

It is very important to keep in mind just what these estimates are – with the main point being that they are estimates and thus uncertain. Actual DNA from actual relatives will, with high probability, yield an actual result somewhere within the bounds of the coverage range. But there are important things to keep in mind when dealing with theoretical situations as we are with any algorithmic estimation. I have put together several relevant points in Appendix A, "Estimates Based on Averages".

Full Coverage Excel Visual Basic Program

My prior paper provided only pseudo-code. And what is shown above provides the pseudo-code extension of the algorithm to full coverage of both the lower and upper bounds. However, in this paper, I want to understand the behavior of the bounds and the propagated average. So, I needed to write a program in some language to calculate both the lower and upper bounds for any configuration of a target ancestor and their DNA-tested descendants.

Here are the reasons that I decided to implement the program in Excel Visual Basic, using a GEDCOM file as input.

1. I wanted a program that anyone can use with their own GEDCOM file to calculate the upper and lower bounds of DNA coverage of anyone in their file who has DNA-tested descendants. In particular, I did not want to use a language that required downloading any software.
2. I wanted something that is completely transparent, with relatively simple code that is relatively easy to understand. I did not want to complicate things by using a language that starts array counting at zero instead of one.
3. What I would really like is for every family tree software product to implement calculation of DNA coverage as a standard feature. It is not enough for Legacy Family Tree to have webinars about DNA coverage. They really should implement it in their Legacy Family Tree software, as should every other family tree software. The Visual Basic program makes it extremely obvious how very simple it is to implement robust DNA coverage estimate calculation.

The full details of the Excel Visual Basic program, including the code itself, are in Appendix B. Appendix C presents a module by module overview of the program.

Behavior of the Bounds and Averages

While my prior paper showed that all but recent ancestors require ranges and not single numbers to accurately report their DNA coverage, I really wanted to run test cases to see how the bounds and averages behave in different scenarios – different configurations of a target ancestor and their DNA-tested descendants in the family tree structure.

The Mama Pepa Scenario

The primary motivation for my deep dive into DNA Coverage has been a project to reconstruct the DNA of Josefa Ruiz, known to her descendants as Mama Pepa. Many of her descendants have DNA-tested. So, I really wanted to see how well they cover Mama Pepa's DNA.

Here is the complete tree. The lowest level person(s) in each branch has done an autosomal DNA test. I have highlighted the test takers in bold underlined red.

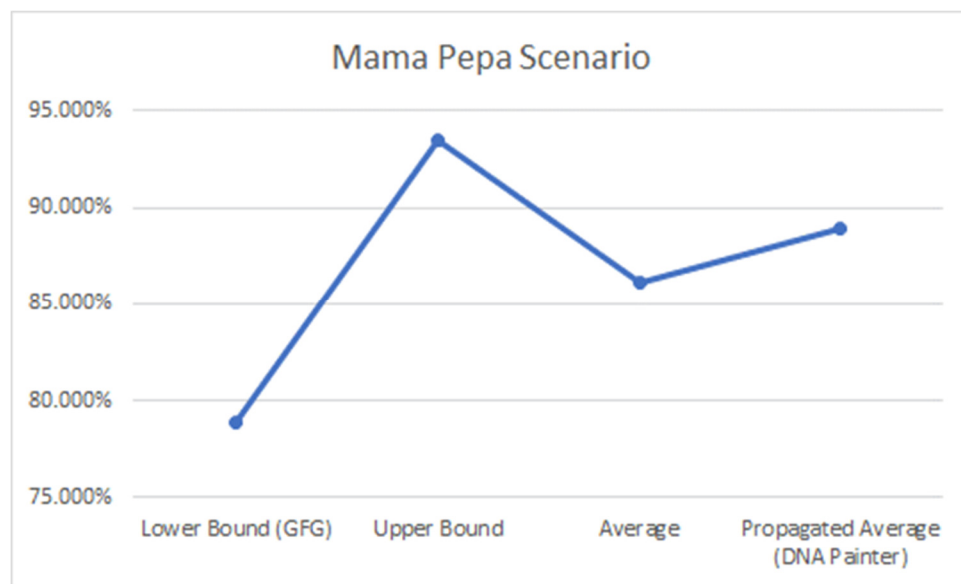
1-Josefa Ruiz	-----4-AdalbertaCh1Ch2
-----2-Rita Salazar	-----4-AdalbertaCh1Ch3
-----3-RitaDau1	-----5-AdalbertaCh1Ch3Ch1
-----4-RitaGrCh1	-----6-AdalbertaCh1Ch3Ch1Ch1
-----5-RitaGrGdChild1	-----3-AdalbertCh2
-----3-RitaDau2	-----4-AdalbertaCh2Ch1
-----4-RitaGdDau2	-----5-AdalbertaCh2Ch1Ch1
-----3-RitaDau3	-----3-AdalbertaCh3
-----4-RitaGdDau3	-----4-AdalbertaCh3Ch1
-----5-RitaGrGdDau3	-----3-AdalbertaCh4
-----2-Adalberta Salazar	-----4-AdalbertaCh4Ch1
-----3-AdalbertaCh1	-----5-AdalbertaCh4Ch1Ch1
-----4-AdalbertaCh1Ch1	-----2-Amalia Salazar

<pre> ----3-AmaliaCh1 ----4-AmaliaCh1Ch1 ----5-AmaliaCh1Ch1Ch1 ----2-Mercedes Salazar ----3-MercedesCh1 ----4-MercedesCh1Ch1 ----5-MercedesCh1Ch1Ch1 ----5-MercedesCh1Ch1Ch2 ----4-MercedesCh1Ch2 ----5-MercedesCh1Ch2Ch1 ----4-MercedesCh1Ch3 ----3-MercedesCh2 ----4-MercedesCh2Ch1 ----5-MercedesCh2Ch1Ch1 ----4-MercedesCh2Ch2 ----4-MercedesCh2Ch3 ----3-MercedesCh3 ----4-MercedesCh3Ch1 ----3-MercedesCh4 </pre>	<pre> ----4-MercedesCh4Ch1 ----3-MercedesCh5 ----3-MercedesCh6 ----3-MercedesCh7 ----4-MercedesCh7Ch1 ----3-MercedesCh8 ----4-MercedesCh8Ch1 ----2-Maria Salazar ----3-MariaCh1 ----3-MariaCh2 ----4-MariaCh2Ch1 ----5-MariaCh2Ch1Ch1 ----3-MariaCh3 ----3-MariaCh4 ----4-MariaCh4Ch1 ----4-MariaCh4Ch2 ----5-MariaCh4Ch2Ch1 ----2-Manuel Salazar ----3-ManuelCh1 Salazar </pre>
--	---

The 27 DNA-tested descendants range from 3 to 6 generations of descent from Mama Pepa through six of her children, some of whom have sizable numbers of tested descendants so that their own DNA coverage is high.

Here are the results.

	Rita	Adalberto	Amalia	Mercedes	Maria	Manuel
Lower	34.375%	56.250%	12.500%	92.773%	84.375%	50.000%
Upper	50.000%	73.633%	12.500%	98.669%	85.938%	50.000%



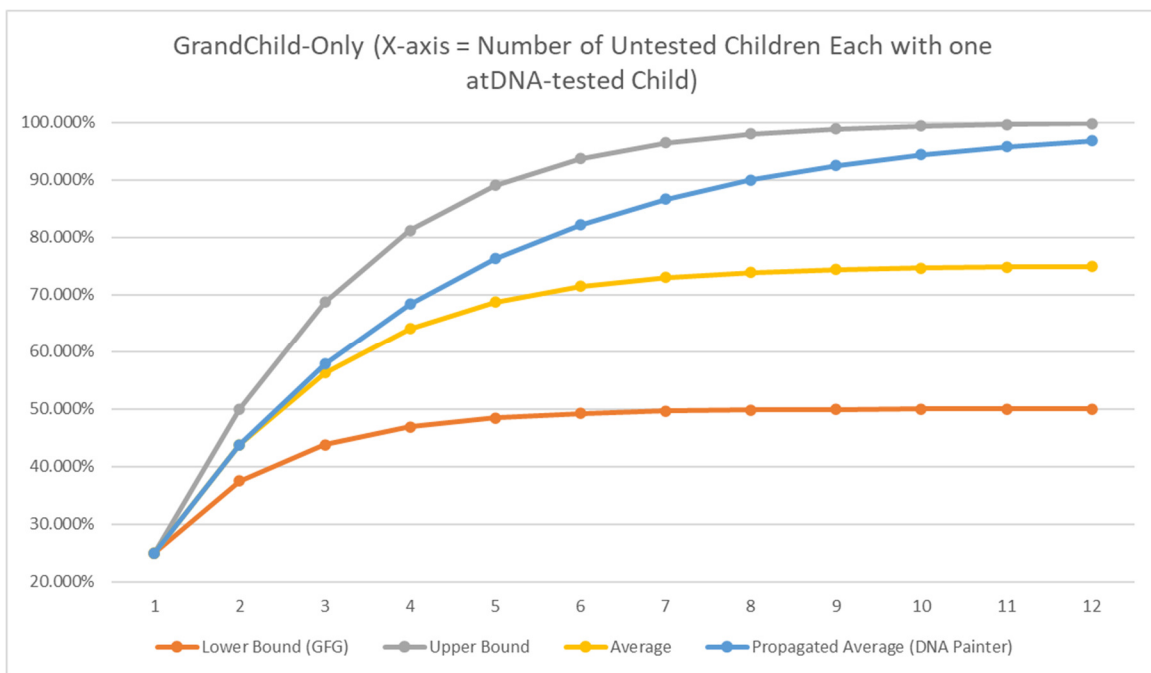
Lower Bound (GFG)	Upper Bound	Average	Propagated Average (DNA Painter)
78.906%	93.442%	86.174%	88.945%

This result aligns with what I expected prior to any of the testing. The lower bound and upper bound are relatively close, making a range of only 14.5%. And the average of the bounds is close to the propagated average. The closer bounds force convergence of the averages.

Even though the propagated average (implemented in DNA Painter) is a bit more optimistic than the average of the bounds, they really are not that far apart. And even though the lower bound (implemented in Graphs for Genealogists) is more pessimistic than the average, they differ only by 7.2%.

Grandchild-Only Scenario

In the first of two suites of stepwise tests, the target person has children who have not done a DNA test but who each have one child who has done a DNA test. Here are the results.



Tested Grand Children	Lower Bound (GFG)	Upper Bound	Average	Propagated Average (DNA Painter)
1	25.000%	25.000%	25.000%	25.000%
2	37.500%	50.000%	43.750%	43.750%

3	43.750%	68.750%	56.250%	57.813%
4	46.875%	81.250%	64.063%	68.359%
5	48.438%	89.063%	68.751%	76.270%
6	49.219%	93.750%	71.484%	82.202%
7	49.609%	96.484%	73.047%	86.652%
8	49.805%	98.047%	73.926%	89.989%
9	49.902%	98.926%	74.414%	92.492%
10	49.951%	99.414%	74.683%	94.369%
11	49.976%	99.683%	74.830%	95.776%
12	49.988%	99.829%	74.909%	96.832%

Consideration of the Bounds

I found this surprising, since I expected the lower bound and the upper bound to converge. Instead of converging, the lower bound asymptotically approaches 50% and the upper bound 100%. So, their average approaches 75%.

In hindsight, this makes perfect sense. The lower bound is the contribution of only one child in combinations while the upper bound is the aggregated contribution of all the children. None of the untested children of the target person will ever have more than 50% coverage. So, using just one of them to calculate the lower bound will never have more than 50% coverage for the lower bound of the target person.

Consideration of the Propagated Average and Lower Bound

The propagated average is overly optimistic (just as the lower bound is overly pessimistic). The propagated average always exceeds the average of the upper and lower bounds and thus increasingly converges on the upper bound while diverging from the lower bound and from the average of the bounds. So, the propagated average implicitly assumes that a best-case scenario is more likely than a worst-case scenario. The expectation, however, is that both the worst-case scenario and the best-case scenario are equally likely so that their average is the most likely overall scenario.

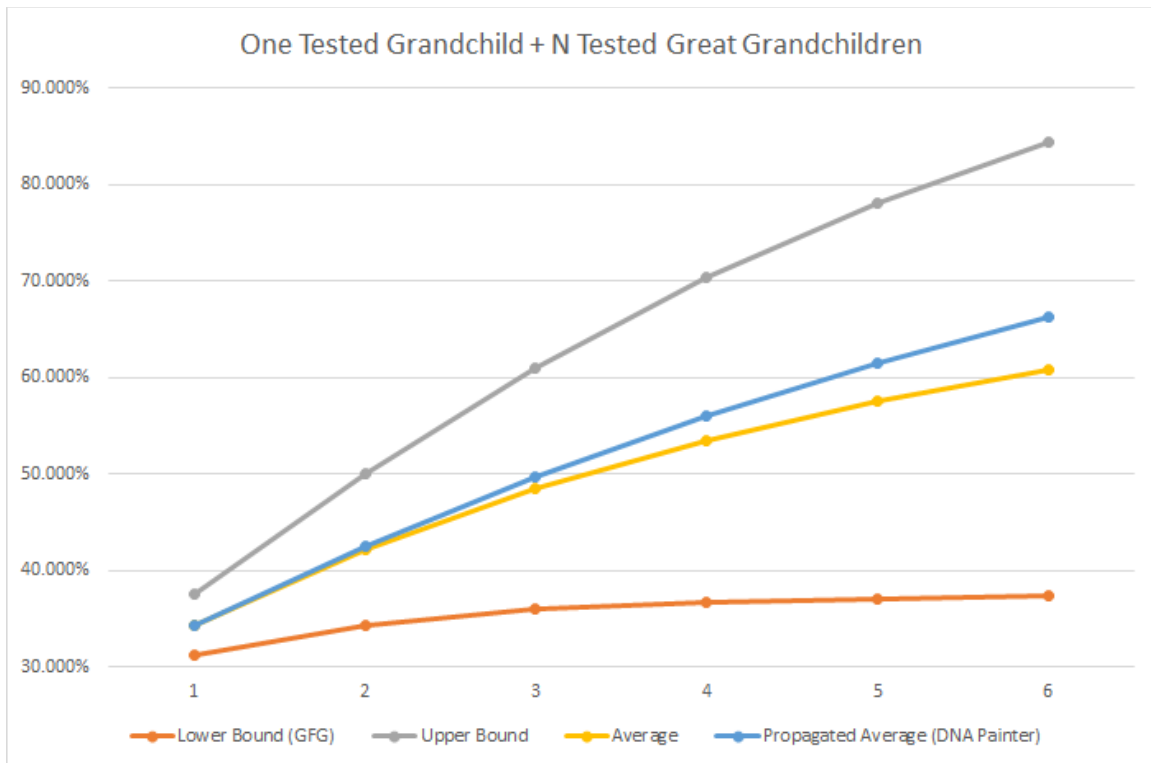
While the propagated average for the scenarios with four or fewer children is at or close to the average of the bounds, with 5 or more children the propagated average starts at 11% more than the average of the bounds and diverges from there so that by 12 children, the propagated average is 29.3% above the average of the bounds and only 3% below the upper bound.

Taking into account the Mama Pepa scenario and the grandchild-only scenario, the behavior of the propagated average and of the lower bound both prove most useful as single-number estimates when the target person has children who either have tested or whose own DNA coverage is high.

One Tested Grandchild + N Tested Great Grandchildren

In the second suite of stepwise tests, the target person has one untested child who has one tested child (the target person's grandchild) plus one or more untested children of the target person who each has one untested child who each has one tested child (a great grandchild of the target person).

Here are the results.



Tested Great Grand Children	Lower Bound (GFG)	Upper Bound	Average	Propagated Average (DNA Painter)
1	31.250%	37.500%	34.375%	34.375%
2	34.375%	50.000%	42.188%	42.578%
3	35.938%	60.938%	48.438%	49.756%
4	36.719%	70.313%	53.516%	56.036%
5	37.109%	78.125%	57.617%	61.532%
6	37.305%	84.375%	60.840%	66.340%

The lower bound asymptotically approaches 37.5%. This is as expected since the one tested grandchild provides 25% coverage of the target person in each case while the combined coverage from the tested grandchildren asymptotically approaches 12.5%,

The upper bound still approaches 100% since it is based on the aggregate of all the test takers.

The propagated average still skews to the optimistic side but does not diverge as rapidly from the average of the bounds since the one tested grandchild moderates it.

An Intuitive Understanding

The way that I find the most intuitive for thinking about the scenarios is to focus on the parent-child family group of the targeted ancestor as the parent. We know that one DNA-tested child covers 50% of the parent, two tested children cover 75%, and N tested children cover $100 - (100/(2^N))$ percent of the parent's autosomal DNA.

The intuitive leap is that situations in which a child's own coverage approaches 100% make the situation approximate the child having done a DNA test.

The key is the impact of the child's coverage on the lower bound. When you can raise the lower bound so that it is closer to the upper bound, all the estimates are squeezed into a smaller range. And you can do that if more of the children are at or near 100% coverage themselves. **For targeted testing, the best new person is the one who will raise the lower bound the most.**

For configurations with only a few children of the target person, using the propagated average in DNA Painter's Coverage Estimator is good. If the number of children increases beyond 4 without the children having a good amount of coverage themselves, then the propagated average becomes overly optimistic. Similarly, the Graphs for Genealogists use of only the lower becomes overly pessimistic in the same scenarios. In such cases, you really do need to know the lower and upper bounds and their average.

But if you have children of the target person whose own coverage is high, then the lower bound increases so that all the estimates become closer to each other.

DNA Coverage as a Standard Feature of Family Tree Software

Autosomal DNA testing began in 2007. It is now 2023. More than twenty million people now have the taking of of an autosomal DNA test as an actual life event. Yet, none of the family tree software companies have a standard "Autosomal DNA Test" life event. It is long past time for the family tree software companies (and the GEDCOM standard) to have a standardized "Autosomal DNA Test" life event.

Now that the full calculation of autosomal DNA coverage estimation is shown to be easily calculated with a relatively simple program, the family tree software companies should provide standard reports of the lower and upper bound estimates of DNA coverage of anyone in the database.

Appendix A: Estimates Based on Averages

First and foremost, keep in mind that the calculated DNA coverage provides only an estimate and not an exact result. The output depends on the input. Only the raw DNA of the testers can provide exact results.

Several factors make these estimates in the right ballpark but not exact.

Average Inheritance

The estimates use the average amount that a person inherits from a specific parent: 50%. The Shared cM project^{vii} by Blaine Bettinger (on the DNA Painter website) uses actual data from known relationships to see the actual range of values of shared centiMorgans for each relationship. While a child inherits on average 3,485 cM from a specific parent, the actual range is 2,376 to 3,720 cM. If we assume that the average is 50% (and thus that the total cMs of a person are 6,970), then the range is from 34% to 53%.

Testing Company Differences: 50% of What?

Briton Nicholson has focused on two more issues. One of those is the differences that the testing companies have for total number of cMs of a person.^{viii} Here are the totals he cites.

Site	Half (cM)	Full (cM)
GEDmatch	3,587	7,174
23andMe	3,537	7,074
MyHeritage	3,500	7,000
AncestryDNA	3,475	6,950
FTDNA	3,384	6,768

The Shared cM Project average of 3,485 cM differs least from the AncestryDNA count. But it ranges from 51.5% (for Family Tree DNA) to 48.6% (for GEDmatch). So, even the Shared cM Project average is not average, depending on which company's results you use.

The algorithm used in this present document does not account for these testing company differences and uses 50%.

Gender Inheritance Differences

Briton Nicholson also pointed out that the transmission of autosomal DNA varies depending on the gender of the parent.^{ix} Here is what he wrote about a simple model he made for autosomal DNA inheritance that deals with the same issue we face with estimating DNA coverage: "For this model, a simple assumption is made that men and women pass DNA to their children in the same manner. In reality, DNA from fathers recombines at lower rates than from mothers. So the variability in shared DNA could be greater from fathers, especially for successively all-male lines. Not taking into account the sex of the parent makes the model vastly simpler to implement and still provides useful approximations, especially for lines that are a fairly even mix of males and females."

The algorithm used in this present document does not account for these gender inheritance differences.

Practicalities of Ancestor DNA Reconstruction

Kevin Borland created Borland Genetics to go beyond theoretical calculations of coverage of an ancestor's DNA by descendants. With Borland Genetics, you can reconstruct an actual DNA kit of the ancestor's DNA from the DNA of the descendants. Kevin Borland in 2019 explained why actual reconstruction of a DNA kit of an ancestor will not result in as much DNA as the theoretical coverage calculations, especially in cases with unphased test result.^x

Dr. David Stumpf has implemented "in-silico" reconstruction in Graphs for Genealogists of an ancestor's DNA from the actual results of tested descendant.^{xi}

Overall Impact on DNA Coverage Estimates

The overall impact on DNA coverage estimates based on the 50% average of DNA inherited by a child from a specific parent is in the right ballpark but should not be taken as an exact percentage. Even the estimates of lower and upper bounds are not hard-line bounds since the Shared cM Project results show that actual cases exist with cMs less than and more than the 50% average.

Appendix B: Full Coverage Excel Visual Basic Program

What the Program Does

You provide the program with two inputs: a GEDCOM file and the RIN (Record Identification Number) of the target person in the GEDCOM file. The program then reads the GEDCOM file and then steps through the family tree starting with the target person and going one generation at a time to identify all DNA-tested descendants of the target person. The program then propagates the children's lower and upper bounds of DNA coverage back up one generation at a time until the target person's lower and upper bound DNA coverage can be calculated. At the end of processing, it populates the Excel worksheet "Results Report" with the complete report, generation by generation, of the bottom-up calculation of the DNA coverage of every person on the line between the target person and the test taker.

Preparing your GEDCOM file

No family tree software company currently has an event to indicate that a person has taken an autosomal DNA test. Thus, you will have to use the family tree software in which you have your database so that you can create a user-defined event which the program will look for as "atDNA".

Here is how I have done this in Legacy Family Tree software.

In the "individual's Information" window for a person, click on the "Add" button to add an event just as you would to add any other life event.

Individual's Information-Josefa Ruiz

Josefa Ruiz

Given:

Surname:

Title Pre.: Title Suf.:

Born: in

Chr: in

Died: in

Buried: in

Living? ☒ Yes ☐ No

☐ M ☒ E ☐ ?

Save

Cancel

Help

▲

▼

⌵

⌶

Events/Facts

Age	Event/Fact	Date	Desc/Place/Notes

+

+

+

+

+

+

+

Add

Edit

Options

Share

Set Order ▲ ▼

Repeat

Exclude from Potential Problems...

User ID AFN FamilySearch ID Find a Grave ID

☐ This individual had no known marriage and no known children

Privacy Settings...

(Not Private)

☐ Birthday Reminder

This opens the “Add Event” window where you need to click on the down arrow by the empty “Event/Fact” pane.

This opens the Master Event Definition List where you need to click on “Add”.

Event Definition (63)	Tag
Adoption	<input type="checkbox"/>
Alt. Birth	<input type="checkbox"/>
Alt. Burial	<input type="checkbox"/>
Alt. Christening	<input type="checkbox"/>
Alt. Death	<input type="checkbox"/>
Alt. Marriage	<input type="checkbox"/>
Annulment	<input type="checkbox"/>

This opens the “Add/Edit an Event Definition” window where you type “atDNA” into the Event Name pane and then click Save.

Add/Edit an Event Definition

Enter an Event Name (up to 100 characters)

atDNA

☒ Show a Description Field
 ☒ Add Notes to sentences
 ☐ Private Event

☒ Show a Date Field
 ☐ Exclude from Potential Problems report

☒ Show a Place Field

[Save](#)
[Cancel](#)
[Help](#)

Event Sentences Roles for those Sharing this Event

How would you like the sentences to read for this type of event?

If All Fields filled:
 If only Date filled:
 If only Place filled:
 If only Description filled:
 If only Desc and Date filled:
 If only Desc and Place filled:
 If only Date and Place filled:
 If Desc, Date and Place are empty:

Replaceable fields you can use within the sentence above:

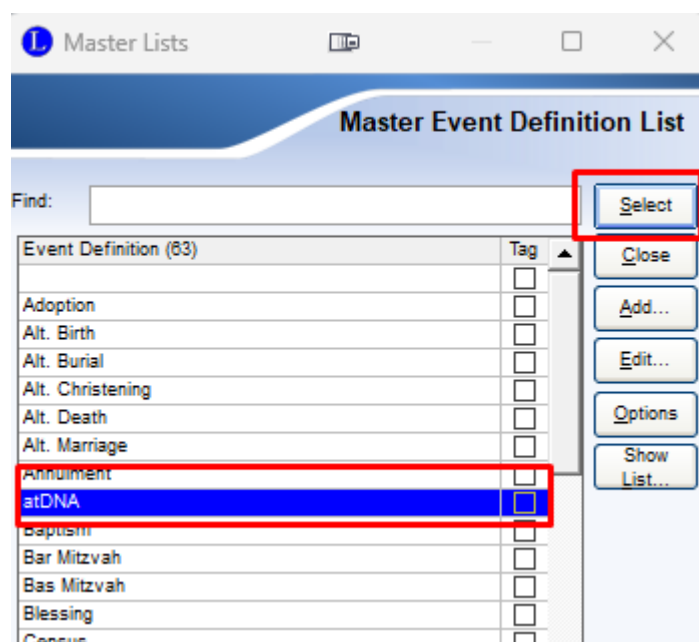
[Name] - full name of individual

[Sentence Preview \(with sample data\)](#)

Options: Male individual event

atDNA: {description}, 28 May 1904, Seattle, King, Washington, United States.* These are the event notes.

Now when you go back to the Master Event Definition List, you will see “atDNA” as an event that you can click on and then click “Select”.



What this will do when you export your database to a GEDCOM is to create the following line in the GEDCOM for this person.

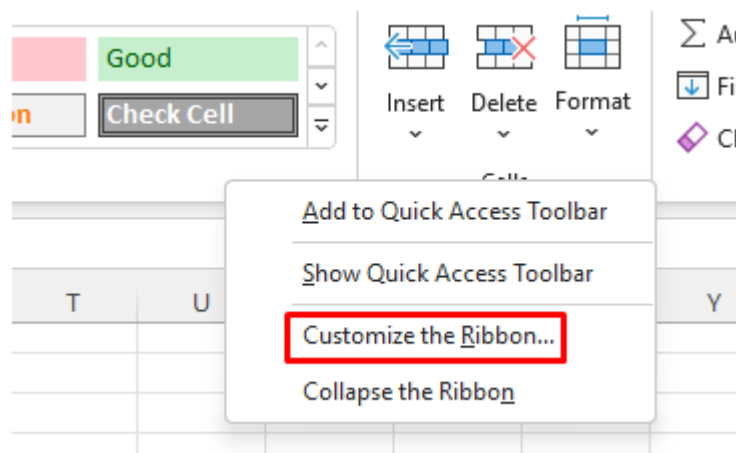
```
1 EVEN
2 TYPE atDNA
```

The Excel Visual Basic Program searches for the “2 TYPE atDNA” record in the GEDCOM file to identify which people in your database have done an autosomal DNA test. So, you need to enter the atDNA event for all those in your database who have done an autosomal DNA test.

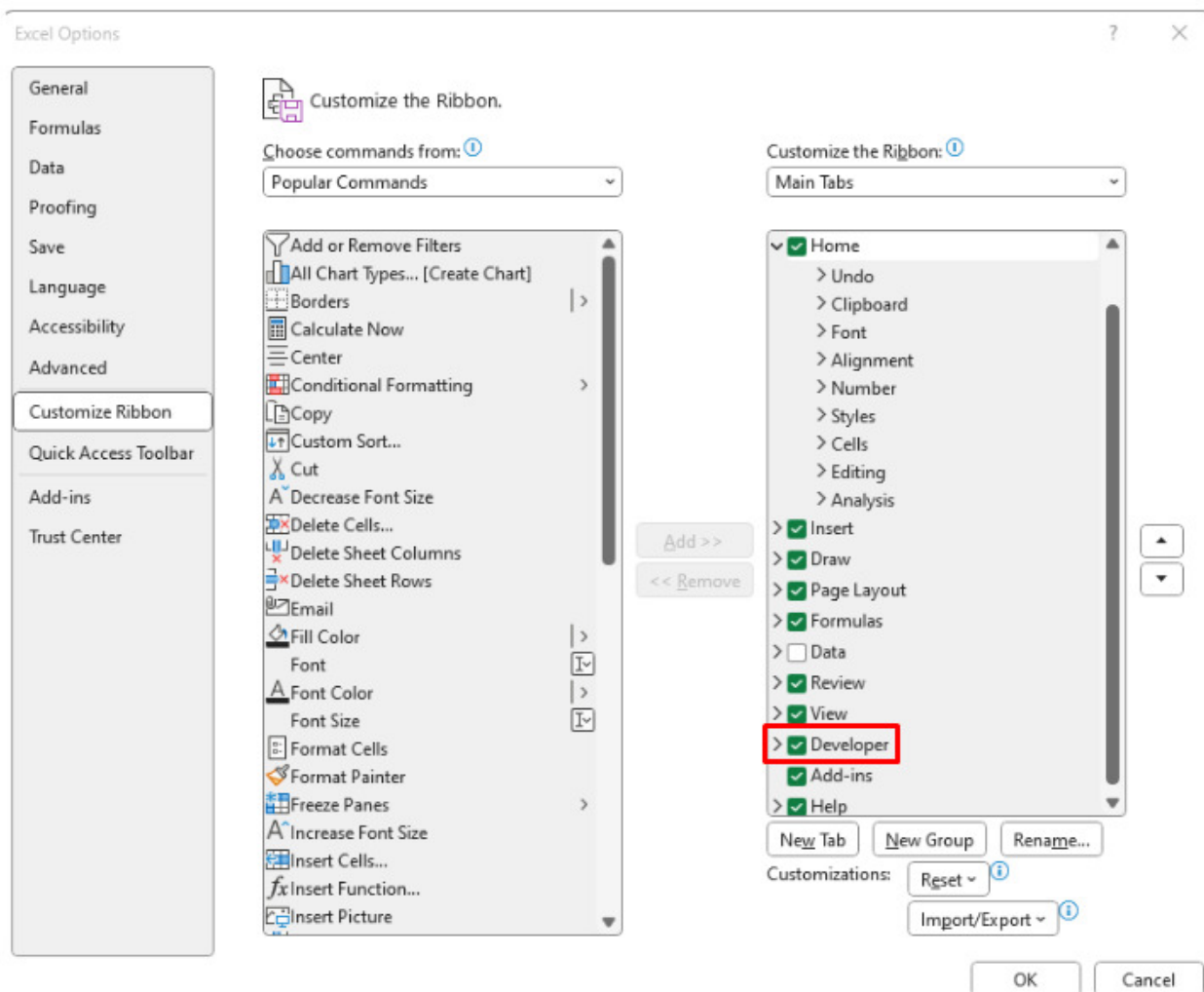
Enabling Visual Basic in Your Excel Spreadsheet

The program runs in Excel. However, you must first enable the Developer tab in Excel. Start with a blank Excel spreadsheet, and name the active worksheet “Results Report”.

In Excel, right click on the ribbon at the top of the spreadsheet and click on “Customize the Ribbon...”.



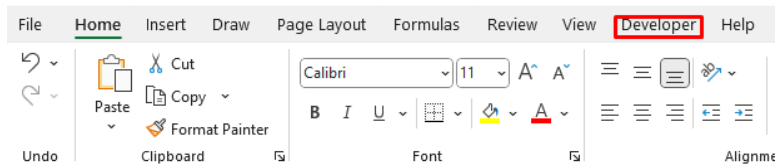
This will open the Excel Options window where you simply want to click on the check box to enable “Developer”. Then click “OK”.



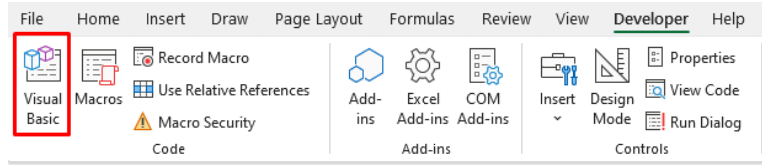
This completes the setup of the Developer tab and its functions for you in Excel.

Setting Up the Visual Basic Program in Excel

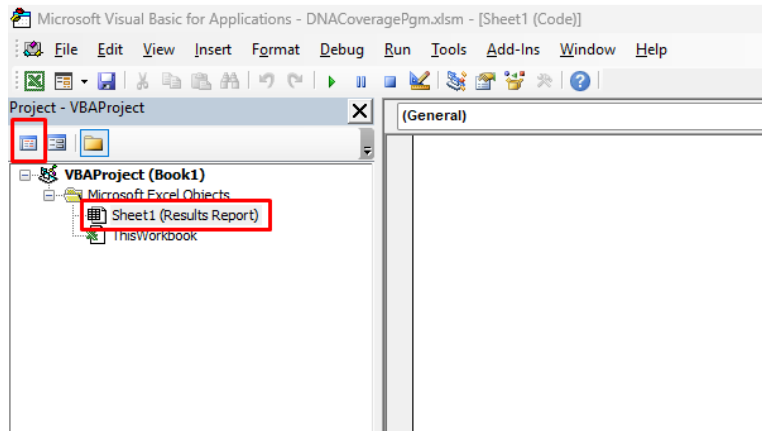
When you now see your ribbon at the top of your Excel worksheet, click on the “Developer” tab.



This will open the Developer tools ribbon where you click on “Visual Basic” to open the Visual Basic editor.



This will open a completely new window “Microsoft Visual Basic for Applications”. Since you already named the worksheet in your main window “Results Report”, this is what the new window looks like.



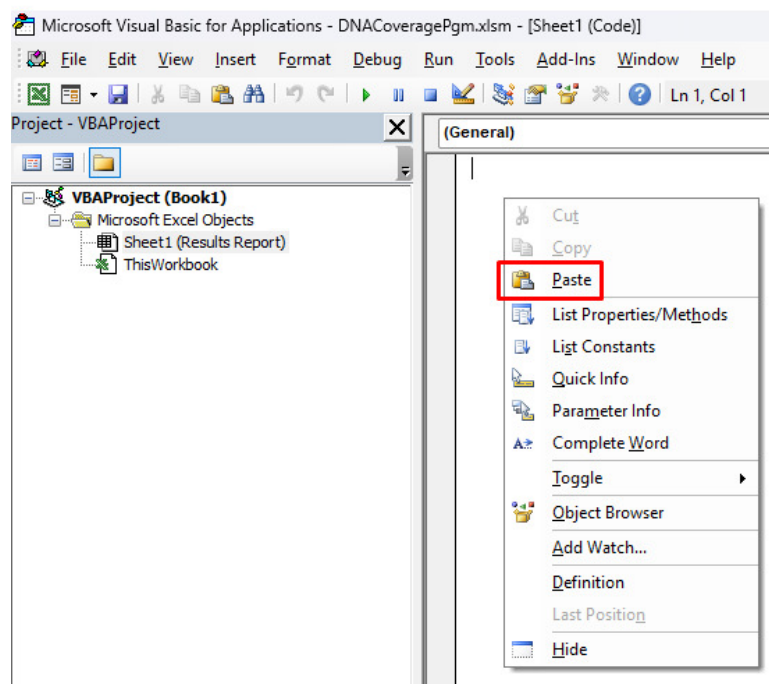
Click on “Sheet1 (Results Report)” and then on the leftmost icon above it. This will change the right-side pane from the gray background to what is shown in the above image.

What you want to do now is to copy the complete text of the program from this paper and paste it into the right-side pane. Select all the text of the program code from this paper, starting with the line that reads

And ending with the line that reads

End Function ' CalcLDNACoverage

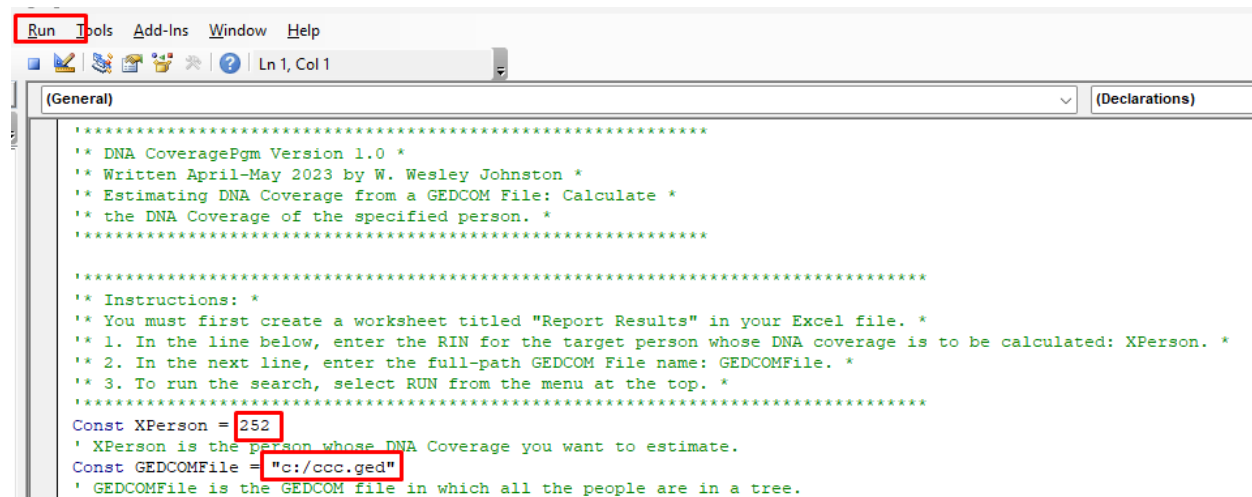
Then right click in the right-side pane of the Visual Basic window and choose the “Paste” option.



Then do a File/Save to save the spreadsheet with the code in it. Since this is now a Visual Basic spreadsheet, Windows will save it as filetype XLSM and not the usual XLSX. This way you have completed the setup of the Visual Basic Program.

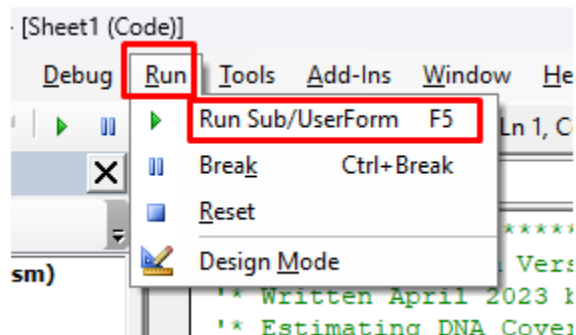
Running the Excel Visual Basic Program

You need to enter the RIN of the target person, as found in your GEDCOM file. And you need to enter the path that tells the program where to find your GEDCOM file. Both of these are near the top of the program code in two different lines.

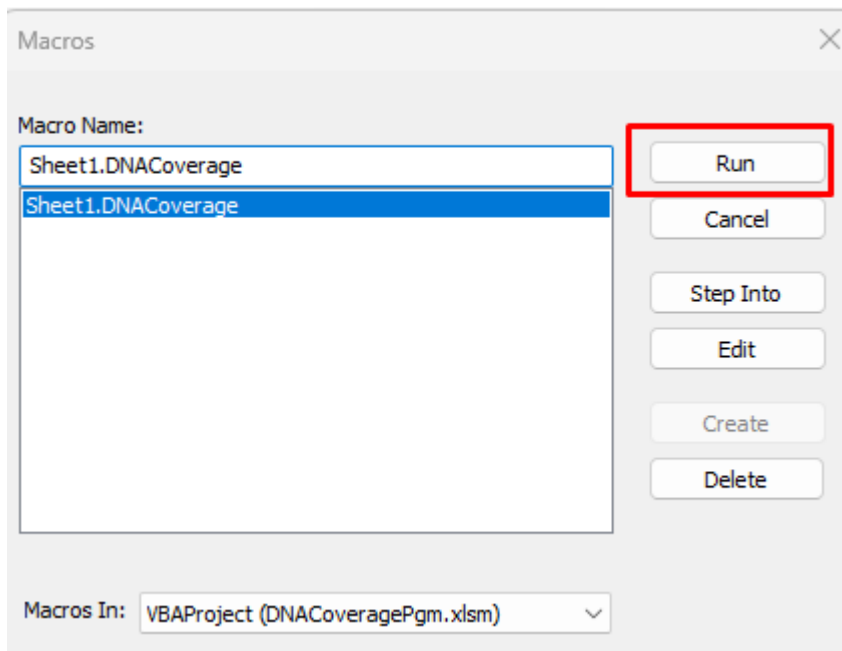


The default setup is to calculate the DNA coverage of the person with RIN 252 in the GEDCOM file that is located at c:/ccc.ged on your computer. So, these are the values you will have to change.

That is all you have to enter. Then go up to the ribbon at the top and click on “Run” and then on “Run Sub/Userform”.



This will open a window labeled “Macros” where you just click again on Run.



The program will run, popping up small windows as it completes milestones. When it says “Done”, the “Results Report” worksheet will display the complete steps in the calculation first of the lower bound and then of the upper bound.

Understanding the Excel Visual Basic Program

If you want to know what is going on “under the hood”, see Appendix C.

The Excel Visual Basic Program Code to Copy

```
*****
'* DNA CoveragePgm Version 1.0 *
'* Written April-May 2023 by W. Wesley Johnston *
'* Estimating DNA Coverage from a GEDCOM File: Calculate *
```

```
'* the DNA Coverage of the specified person. *
'*****

'*****

'* Instructions: *
'* You must first create a worksheet titled "Report Results" in your Excel file. *
'* 1. In the line below, enter the RIN for the target person whose DNA coverage is to be calculated:
XPerson. *
'* 2. In the next line, enter the full-path GEDCOM File name: GEDCOMFile. *
'* 3. To run the search, select RUN from the menu at the top. *
'*****

Const XPerson = 252
' XPerson is the person whose DNA Coverage you want to estimate.
Const GEDCOMFile = "c:/ccc.ged"
' GEDCOMFile is the GEDCOM file in which all the people are in a tree.

'*****

'* Globally define all the needed variables,
'* constants and arrays.
'*

'* The critical element for fast lookups is being able to
'* use an index-value lookup in an array.
'*****

Const RINLimit = 100000, MRINLimit = 50000, FPLimit = 300, PFLimit = 200
' See Table1KidLimit in subroutine for limiting number of children and also the related array size.

Dim RinFlag(1 To 2, 1 To RINLimit) As Boolean ' Cell Value is TRUE or FALSE
Dim RinName(1 To RINLimit) As String ' Cell Value is the Person's name
Dim RinatDNA(1 To RINLimit) As Boolean ' Cell Value is the Person's atDNA-tested status
Dim NRINS As Integer ' Count of the number of RINs in the file
Dim MAXRIN As Integer ' The highest RIN number in the file
Dim Rin As Integer
Dim MRin As Integer

'*****

'FamPers is a table of all the persons in a specific family.
'Each person (RIN) is in a separate record for the family (MRIN).
'So, a family is a collection of (MRIN, RIN) pairs in the array.
'*****

' FamPers (or just FP) uses a family MRIN to find all members' RINs
Dim FamPers(1 To MRINLimit, 1 To FPLimit) As Long ' Cell Value is a +/- RIN
'*****

'PersFam is a table of all the families to which a specific person belongs.
'Each family (MRIN) is connected to each child (RIN) in the family.
'So, a person is a collection of (RIN, MRIN) pairs in the array.
'*****
```

Dim PersFam(1 To RINLimit, 1 To PFLimit) As Long ' Cell Value is +/- MRIN

' - = Child (i.e., if -MRin or -Rin, then person is a child in that MRin)

Dim MAXMRIN As Integer ' The highest MRIN number in the file

Dim TargetUDNACoverage As Double

Dim PFSpouse As Boolean

Dim XYCt As Integer

Dim LeftSide As String

```
PrepEndTime = Time
```

```
showit = "DONE WITH PREP--Start-" & StartTime & " PrepEnd-" & PrepEndTime & " NRINS =" & NRINS &
" MAXRIN =" & MAXRIN & " NMRINS =" & NMRINS & " MAXMRIN =" & MAXMRIN
MsgBox showit
```

```
'*****
*****
' Write the first output line to the OutputRow array and increment the line counter.
'*****
*****
```

```
LineCount = 1
OutputRow(LineCount) = "Calculation of the DNA Coverage of RIN " & XPerson & " (" &
RinName(XPerson) & ")"
LineCount = LineCount + 1
```

```
'*****
*****
' Calculate the DNA Coverage of the target person.
'*****
*****
```

```
TargetLDNACoverage = -1
TargetLDNACoverage = CalcLDNACoverage(XPerson) ' This is the real engine of the program for lower
bound calculation.
showit = "DONE WITH Lower Bound"
MsgBox showit
```

```
TargetUDNACoverage = -1
TargetUDNACoverage = CalcUDNACoverage(XPerson) ' This is the real engine of the program for lower
bound calculation.
showit = "DONE WITH Upper Bound"
MsgBox showit
showit = "1-LineCount=" & LineCount
MsgBox showit
```

```
'*****
*****
' Now that the DNA Coverage is calculated, write all the lines of output to the Results Report worksheet.
'*****
*****
```

```
Set wout = Sheets("Results Report")
wout.Activate
wout.Cells.ClearContents
```

```
' Writing output now
showit = "2-LineCount=" & LineCount
```

```
MsgBox showit
For j = 1 To LineCount
wout.Cells(j, 1) = OutputRow(j)
Next ' For j = 1 to LineCount
```

```
RunEndTime = Time
showit = "DONE"
MsgBox showit
```

```
'*****
'* END OF MAIN PROGRAM *
'*****
```

```
End Sub
Function InitVars()
```

```
NRINS = 0
MAXRIN = 0
NMRINS = 0
MAXMRIN = 0
```

```
For LoopCt = 1 To RINLimit
RinName(LoopCt) = ""
RinatDNA(LoopCt) = False ' Intialize the atDNA-test status as False for everyone
```

```
For LoopCt2 = 1 To PFLimit
PersFam(LoopCt, LoopCt2) = 0
Next ' For LoopCt2 = 1 To PFLimit
If LoopCt < MRINLimit + 1 Then
For LoopCt2 = 1 To FPLimit
FamPers(LoopCt, LoopCt2) = 0
Next ' For LoopCt2 = 1 To FPLimit
End If ' If LoopCt < MRINLimit + 1 Then
Next ' For LoopCt = 1 To RINLimit
```

```
UnableToExpandLevel = False
FoundAConnection = False
```

```
End Function
Function ReadGEDCOM()
```

```
'*****
'* Read through the GEDCOM file and
'* 1. Define RIN-Name Index -- OBSOLETE: and X and Y RIN Flag Arrays
'* 2. Define Family-Person File and Family Counter Array
'* Only the INDI records of the GEDCOM file are used.
'*****
```

Const ForReading = 1, ForWriting = 2, ForAppending = 3

Const TristateUseDefault = -2, TristateTrue = -1, TristateFalse = 0

Dim fso, ts, s

Set fso = CreateObject("Scripting.FileSystemObject")

Set ts = fso.OpenTextFile(GEDCOMFile, ForReading, TristateUseDefault)

Dim RINSDone As Boolean

RINSDone = False

Dim EndofINDI As Boolean

EndofINDI = False

Dim PersCount As Integer ' Will run from 1 to FPLimit

Dim FamCount As Integer ' Will run from 1 to PFLimit

'Dim loopct As Integer

'For loopct = 1 To 500

' *

' ***** Start Main GEDCOM Read Routine *****

' *

Do While ts.AtEndOfLine <> True ' Read all the records in the GEDCOM file

s = ts.ReadLine ' This reads the current line of the GEDCOM file and assigns it to the variable s.

' *

' ***** Start Section FoundZeroINDI *****

' * Find a "0 ... INDI" record which is the first record of Person X.

' * Capture the RIN number, the name, the atDNA flag (if any) and the records of the families to which Person X belongs.

' * Remember that there are "0" records before the first "0 ... INDI". You have to bypass those first.

' *

FoundZeroINDI:

If RINSDone = False Then ' Keep processing until reach the end of the RINs. After that do the MRINs.

If Left(s, 1) = 0 Then ' Look for a zero in column 1.

If Right(s, 4) = "INDI" Then ' Found the initial record for a new person *****

' This begins a lot of processing before you find the associated End If

NRINS = NRINS + 1 ' Increment the count of the number of RINs in the file

' FoundZeroINDI-Part 1 ***** Now find and capture the RIN value itself *****

RightSide = Right(s, Len(s) - 4)

For ctspaces = 1 To 10 ' Find the first @ symbol that defines the RIN number location.

If Mid(RightSide, ctspaces, 1) = "@" Then

GoTo FoundRinAt

End If

Next ' For ctspaces = 1 To 10

FoundRinAt:

' Once a RIN is found, capture the RIN value


```
Rin = (Left(RightSide, (ctspaces - 1)))
```

```
' FoundZeroINDI-Part 2 ***** Now read the next line which is the name line and capture the name
*****
```

```
s = ts.ReadLine
```

RinName(Rin) = Right(s, Len(s) - 7) ' Capture the name for Person X (the Person with this RIN).

! *

' FoundZeroINDI-Part 3 ***** Now, READ ALL THE REST OF THE RECORDS FOR Person X. *****

! *

```
' * Find any TYPE or FAMS or FAMC records for this person and load to arrays. ****'
```

' "2 TYPE atDNA" records are those that indicate that the person has done an autosomal DNA test.

'1 FAMS" records are those where the person is a spouse/parent in the family.

' "1 FAMC" records are those where the person is a child in the family.

' Do not confuse the "1 FAM" records of the person (RIN) with the "0 @F1@ FAM" MRIN records.

' The TYPE record, if there is one, comes before the FAMS or FAMC records.

' So, the initial search is for a TYPE record.

' If you find a new zero-record, then you have found a new person so that you have to start work on that person.

EndofINDI = False

Do Until EndofINDI = True ' Read all the records of person X until done with person X.

```
s = ts.ReadLine
```

' Sending control to LoopToFindFAM causes this Do Loop to end processing for Person X and read the next GEDCOM record.

' First, check to see if you have finished processing Person X.

If $\text{Left}(s, 1) = 0$ Then ' Check to see if you found a new 0 (zero) record which means you are done processing Person X. So, stop processing person X.

EndofINDI = True ' You stop processing Person X by setting EndofINDI to TRUE to end the Do loop which will take you to EndofINDI.

End If ' If Left(s, 1) = 0

1

[illegible]

! *

' FoundZeroINDI-Part 3A ***** See if Person X did an autosomal DNA test. *****

! *

' You will know that they tested if they have a "2 TYPE atDNA" record.

1

[illegible]

If Left(s, 12) = "2 TYPE atDNA" Then ' The person has tested for autosomal DNA. So, set their RinatDNA flag to TRUE.

```
RinatDNA(Rin) = True
```

GoTo LoopToFindFAM ' Since you have processed this record, go on to read the next record.


```
' *
FoundSMRinAt:
MRin = (Left(RightSide, (ctspaces - 1))) ' The MRIN to be captured starts in the right side 1st position
(column 10) and ends in the position just before ctspace.
' Store in the Family-Person Array
For PersCount = 1 To FPLimit
If FamPers(MRin, PersCount) = 0 Then
FamPers(MRin, PersCount) = Rin
GoTo LoadFAMStoPF
End If
' Since this Rin slot is already filled, check the next one.
Next ' For PersCount = 1 To 30
LoadFAMStoPF:
' Store in the Person-Family Array
For FamCount = 1 To PFLimit
If PersFam(Rin, FamCount) = 0 Then
PersFam(Rin, FamCount) = MRin
' showit = "PersFam(" & Rin & "," & FamCount & ") = " & MRin
' MsgBox showit
GoTo LoopToFindFAM ' done processing this FAMS record
End If
' Since this Rin slot is already filled, check the next one.
Next ' For FamCount = 1 To 30
End If ' If Right(s, 4) = "FAMS"

' *****
' FoundZeroINDI-Part 3B2: FAMC Check and Processing
' *****

If Mid(s, 3, 4) = "FAMC" Then
' Write as NEGATIVE number since positive numbers will designate parents and negatives will designate
children in the family.
' Parse the C-MRIN.
'
' The following code starts at the 9th position in the record and searches for @.
' The record is "1 FAMC $F" in the first 9 positions. The MRIN value starts in the 10th position.
' To find the MRIN value, you have to find the second @ in the record.
' You then know the starting and ending positions of the MRIN value so that you can capture it.
RightSide = Right(s, Len(s) - 9)
For ctspace = 1 To 10
If Mid(RightSide, ctspace, 1) = "@" Then
GoTo FoundCMRinAt
End If
Next
' *
' Subroutine FoundCMRinAt to capture the FAMC MRIN
' * There is a hazard here. If any GEDCOM ever has a huge MRIN number, then the processing will fall
through here.
' * That would lead to the subroutine FoundCMRinAt being executed when it should not.
```

' *

FoundCMRinAt:

MRin = (Left(RightSide, (ctspaces - 1))) ' The MRIN to be captured starts in the right side 1st position (column 10) and ends in the position just before ctspace.

' Store in the Family-Person Array

For PersCount = 1 To FPLimit

If FamPers(MRin, PersCount) = 0 Then

FamPers(MRin, PersCount) = -Rin

GoTo LoadFAMCtoPF

End If

' Since this Rin slot is already filled, check the next one.

Next ' For PersCount = 1 To 30

LoadFAMCtoPF:

' Store in the Person-Family Array

For FamCount = 1 To PFLimit

If PersFam(Rin, FamCount) = 0 Then

PersFam(Rin, FamCount) = -MRin

GoTo LoopToFindFAM ' done processing this FAMC record

End If

' Since this Rin slot is already filled, check the next one.

Next ' For FamCount = 1 To 30

End If ' If Right(s, 4) = "FAMC"

LoopToFindFAM:

' This step takes you to the next record in the GEDCOM.

Loop ' Do Until EndofINDI = True

,

' End of FoundZeroINDI-Part 3 and of all processing for Person X.

,

,

' EndofINDI: Finished processing Person X. So, see if it current record is the last RIN record.

,

EndofINDI: ' You have finished the current individual's records.

' The first MRIN record will have the form "0 ... FAM".

' A search for "FAM" in the rightmost 3 characters will tell you that you have reached the MRIN records.

If Right(s, 3) = "FAM" Then ' The first "0 ... FAM" record in the GEDCOM indicates RINS are done.

RINSDone = True

GoTo RINSDone

End If ' If Right(s, 3) = "FAM"

GoTo FoundZeroINDI: ' go back and process the new individual you have found.

Else ' If Right(s, 4) = "INDI" Then

If Right(s, 3) = "FAM" Then ' The first "0 ... FAM" indicates RINS are done.

RINSDone = True

GoTo RINSDone

End If ' If Right(s, 3) = "FAM" Then

End If ' If Right(s, 4) = "INDI" Then

End If ' If Left(s, 1) = 0 Then

Else ' If RINSDone = False Then

End If ' If RINSDone = False Then

,

' RINSDone

' You have finished processing all the RINS in the GEDCOM.

' Capture the NMAXRINS count of RINS and MAXMRIN count of MRINs.

' Then exit the ReadGEDCOM Function.

,

RINSDone:

'Next ' For loopct = 1 To 100

Loop 'Do While ts.AtEndOfLine <> True

ts.Close

MAXRIN = Rin ' Capture the highest RIN, which will be the last one

' Figure out MAXMRIN and NMRINS

For LoopCt = 1 To MRINLimit

If FamPers(LoopCt, 1) <> 0 Then

NMRINS = NMRINS + 1

MAXMRIN = LoopCt

End If ' If FamPers(LoopCt, 1) <> 0

Next ' For LoopCt = 1 To MRINLimit

End Function ' ReadGEDCOM()

Function CalcLDNACoverage(ReceivedRIN)

```
'Initial setup for Target Person
'Find all children of target person and create upper and lower bounds arrays for them --
LTable2(ChildSeq,ChildRIN,ChildCoverage) and UTable2
'Set values of N, P, M and set up LTable1 and LTable2 of Venn Diagram representation for lower and
upper bounds
'For each child, call GetChildCoverage for child and enter it in Table2
'When processed all children, use tables to calculate target person's coverage, report it and end.
```

```
'CalcLDNACoverage:
'Find all children of received person and create upper and lower bounds arrays for them --
LTable2(ChildSeq,ChildRIN,ChildCoverage) and UTable2
'Set values of N, P, M and set up LTable1 and LTable2 of Venn Diagram representation for lower and
upper bounds
'For each child, call GetChildCoverage for child and enter it in Table2
'When processed all children, use tables to calculate received person's coverage, report it for family,
return the coverage to call and end.
```

```
CalcLDNACoverage = 0
Dim RinReceived As Integer
RinReceived = ReceivedRIN
```

```
Dim MRINFound As Integer
Dim KidsFound As Integer
```

```
*****
```

```
' Set up LTable2 and UTable2 (Lower and Upper-Bound Table 2s)
' The index value is the number of the child in the table.
' xTable2RIN is the RIN of the child.
' xTable2Coverage is the DNA coverage of the child, initially zero.
```

```
*****
```

```
Dim LTable2RIN(1 To FPLimit) As Integer
Dim LTable2Coverage(1 To FPLimit) As Double
```

```
*****
```

```
' If the person has DNA-tested, set value to 1 (100%).
' Then exit the function.
```

```
*****
```

```
If RinatDNA(RinReceived) = True Then ' Do I need to write an outputrow for
this????????????????????????????????????????????????????????
```

```
CalcLDNACoverage = 1
GoTo EndCalc ' Go to the exit of this function.
End If
```

```
*****
*****
```

```
' If you reach this point, the person has NOT done an autosomal DNA test.
' So, find Person X's children to calculate the person's coverage.
```



```
*****
*****
```

```
*****
*****
```

```
' To find the children, first find all the PersFam entries for which person X is a parent.
' That is, find all the families in which person X is a parent
' Search PersFam for the RIN of Person X (the row variable of PersFam).
' Then for each child of that MRIN (in FamPers), if the value (the RIN) is positive, Person X is a parent in
that MRIN.
' So, use that MRIN to search FamPers.
' For any FamPers record for this MRIN, if the second element (RIN) is negative, then that person is a
child of that parent in that family.
' So, capture the postive value of that child's RIN into Table 2 (both) as a child of Person X.
```

```
*****
*****
```

```
KidsFound = 0
For CountFamilies = 1 To PFLimit
    If PersFam(RinReceived, CountFamilies) > 0 Then ' Have found a family record (MRIN) of Person X - so
now find all children of that family and load to Table 2
        MRINFound = PersFam(RinReceived, CountFamilies)
        For CountChildren = 1 To FPLimit
            If FamPers(MRINFound, CountChildren) < 0 Then
                KidsFound = KidsFound + 1
                LTable2RIN(KidsFound) = Abs(FamPers(MRINFound, CountChildren))
                LTable2Coverage(KidsFound) = 0 ' Initialize the Child's DNA Coverage at zero
            End If ' If FamPers(MRINFound, CountChildren) <> 0
        Next ' For CountChildren = 1 To FPLimit
    End If ' If PersFam(RinReceived, CountFamilies) <> 0
Next ' For CountFamilies = 1 To PFLimit
```

```
*****
*****
```

```
' So now, you have built both versions (L and U) of Table2.
' So calculate the key constants.
' N = number of children of the parent
' P = Pieces of the Venn diagram of the parent =(2^N)-1
' M = Max percent of parent that each piece of the Venn diagram can contribute = 1/(2^N)
' M is calculated as a decimal and not as a percent
```

```
*****
*****
```

```
Dim N, P As Long
Dim M As Double
```

```
N = KidsFound
P = (2 ^ N) - 1
M = 1 / (2 ^ N)
```

```

*****
*****
' Now create both versions (L and U) of Table1. Each column is a separate 1-dim array.
' Each row of Table 1 has one column for each child with the binary value of that row number in those
columns.
' Each row of Table 1 also has one column for M which is the constant M for all rows.
' Each row of Table 1 also has one column for W (weight)/
' -- For the lower bound W is the maximum coverage of any child with a 1 in their column in that row.
*****
*****
' -- For the upper bound W is the minimum of 1 or of the sum of the coverages of all the children with a
1 in their column in that row.
*****
*****
' Each row of Table 1 also has one column for R (result) which is M * W.
*****
*****

*****
' Set up LTable1 and UTable1 (Lower and Upper-Bound Table 1s)
' The index value is the number of the piece of the Venn diagram in the table.
' Thus the index number is the row number in Table1.
'
*****
Const Table1PieceLimit = 4096 ' Allows for up to 12 children
Const Table1KidLimit = 12 ' Allows for up to 12 children

Dim LTable1Binary(1 To Table1PieceLimit, 1 To Table1KidLimit) As Integer
Dim LTable1M(1 To Table1PieceLimit) As Double
Dim LTable1W(1 To Table1PieceLimit) As Double
Dim LTable1R(1 To Table1PieceLimit) As Double

*****
' Set the Binary Value for Each Piece/Row in the Child Columns
' Convert Venn Diagram piece number to its binary equivalent and put bits into Table1 child cells
' Works by reducing piece number by power of 2 on each iteration
*****
For PieceNum = 1 To P
    LTable1W(PieceNum) = 0
    LTable1M(PieceNum) = M
    LTable1R(PieceNum) = 0
    Remainder = PieceNum
    For KidNum = 1 To N
        BitPower = N - KidNum
        Bit = WorksheetFunction.Power(2, BitPower) ' Power(2, BitPower)
        If Remainder >= Bit Then
            LTable1Binary(PieceNum, KidNum) = 1
            Remainder = Remainder - Bit
        End If
    Next KidNum
Next PieceNum

```

```

Else
    LTable1Binary(PieceNum, KidNum) = 0
End If ' If Remainder >= Bit
Next ' For KidNum = 1 To N
Next ' For PieceNum = 1 To P

'*****
'*****

' Now that both tables are set up, step through each child to obtain their DNA coverage in Table2.
' This is done be recursive calls to this same routine.
'*****
'*****

For ChildCount = 1 To N
    ChildRIN = LTable2RIN(ChildCount)
    TargetLDNACoverage = CalcLDNACoverage(ChildRIN)
    LTable2Coverage(ChildCount) = TargetLDNACoverage
Next ' For ChildCount = 1 To N

'+++++
'+++++
'+++++ THE KEY CALCULATION
'+++++
'+++++
'+++++
' Now the children have all been found and their own DNA Coverage has been calculated.
' So, calculate the coverage of their parent.
' This is the KEY CALCULATION of this function.
' It is the only place that the calculation of the lower and upper bounds differ.
'+++++
'+++++
'+++++
'+++++
'+++++
'+++++

'*****
'*****

' Now that the DNA coverage of all children is known, calculate the DNA coverage of Person X who is
RINReceived.
' Calcualte the contribution of each piece of the Venn Diagram and add them up.
'*****
'*****

Dim MaxChildDNA As Double
MaxChildDNA = 0
Dim ChildProduct As Double
Dim WorkingCalcLDNACoverage As Double
WorkingCalcLDNACoverage = 0

```

```

For PieceCount = 1 To P
'
*****
*****
' Calculate lower bound W = max of the products of each child's coverage (from Table 2)
' Calculate lower bound R = W * M
'
*****
*****
    MaxChildDNA = 0
    For ChildCount = 1 To N
    '
*****
*****
        ' Calculate lower bound W = max of the products of each child's coverage (from Table 2)
        '
*****
*****
            ChildProduct = LTable2Coverage(ChildCount) * LTable1Binary(PieceCount, ChildCount)
            MaxChildDNA = WorksheetFunction.Max(MaxChildDNA, ChildProduct)
            LTable1W(PieceCount) = MaxChildDNA
        Next 'For ChildCount = 1 To N
        LTable1R(PieceCount) = LTable1W(PieceCount) * LTable1M(PieceCount)
        WorkingCalcLDNACoverage = WorkingCalcLDNACoverage + LTable1R(PieceCount)
    Next ' For PieceCount = 1 To P
'
*****
*****
' Set the output value to be returned
'
*****
*****
CalcLDNACoverage = WorkingCalcLDNACoverage
'
*****
*****
' Now that the calculation for this person is complete, write the output rows
' for this person and the children and Tables 1 and 2 values.
'
*****
*****
OutputValue = "-----"
OutputRow(LineCount) = OutputValue
LineCount = LineCount + 1

```

```

OutputValue = "LOWER BOUND DNA Coverage of RIN " & RinReceived & " (" & RinName(RinReceived) &
") = " & CalcLDNACoverage
OutputRow(LineCount) = OutputValue
LineCount = LineCount + 1

```

```

OutputValue = "Children of RIN " & RinReceived & " (" & RinName(RinReceived) & ")"
OutputRow(LineCount) = OutputValue
LineCount = LineCount + 1

```

```

For ChildCount = 1 To N
    OutputValue = "RIN " & LTable2RIN(ChildCount) & " (" & RinName(LTable2RIN(ChildCount)) & ")"
    OutputRow(LineCount) = OutputValue
    LineCount = LineCount + 1
Next ' For ChildCount = 1 to N

```

```

'
*****
*****

```

```

' Write out Table 2

```

```

'
*****
*****

```

```

OutputValue = "***** Table 2 *****"
OutputRow(LineCount) = OutputValue
LineCount = LineCount + 1

```

```

OutputValueRIN = ""
OutputValueCoverage = ""
For ChildCount = 1 To N
    OutputValueRIN = OutputValueRIN & LTable2RIN(ChildCount) & "--"
    OutputValueCoverage = OutputValueCoverage & LTable2Coverage(ChildCount) & "--"
Next ' For ChildCount = 1 to N

```

```

OutputRow(LineCount) = "RINS--" & OutputValueRIN
LineCount = LineCount + 1

```

```

OutputRow(LineCount) = "DNA--" & OutputValueCoverage
LineCount = LineCount + 1

```

```

'
*****
*****

```

```

' Write out Table 1

```

```

'
*****
*****

```

```

OutputValue = "***** Table 1 *****"
OutputRow(LineCount) = OutputValue

```

LineCount = LineCount + 1

OutputValuePiece = ""

OutputValueHeader = "----R----M----W<<<<"

For ChildCount = 1 To N

OutputValueHeader = OutputValueHeader & LTable2RIN(ChildCount) & "+"

Next ' For ChildCount = 1 To N

OutputRow(LineCount) = OutputValueHeader

LineCount = LineCount + 1

OutputValuePiece = ""

For PieceCount = 1 To P

OutputValuePiece = OutputValuePiece & "----" & LTable1R(PieceCount) & "--" &

LTable1M(PieceCount) & "--" & LTable1W(PieceCount) & "<<<<"

For ChildCount = 1 To N

OutputValuePiece = OutputValuePiece & LTable1Binary(PieceCount, ChildCount) & "+"

Next ' For ChildCount = 1 to N

OutputRow(LineCount) = OutputValuePiece

LineCount = LineCount + 1

OutputValuePiece = ""

Next ' For PieceCount = 1 to P

EndCalc:

End Function ' CalcLDNACoverage

Function CalcUDNACoverage(ReceivedRIN)

```
'+++++
+++++
+++'
```

'Initial setup for Target Person

'Find all children of target person and create upper and lower bounds arrays for them --

LTable2(ChildSeq,ChildRIN,ChildCoverage) and UTable2

'Set values of N, P, M and set up LTable1 and LTable2 of Venn Diagram representation for lower and upper bounds

'For each child, call GetChildCoverage for child and enter it in Table2

'When processed all children, use tables to calculate target person's coverage, report it and end.

'CalcLDNACoverage:

'Find all children of received person and create upper and lower bounds arrays for them --

LTable2(ChildSeq,ChildRIN,ChildCoverage) and UTable2

'Set values of N, P, M and set up LTable1 and LTable2 of Venn Diagram representation for lower and upper bounds

'For each child, call GetChildCoverage for child and enter it in Table2

'When processed all children, use tables to calculate received person's coverage, report it for family, return the coverage to call and end.

CalcUDNACoverage = 0

Dim RinReceived As Integer

RinReceived = ReceivedRIN

Dim MRINFound As Integer

Dim KidsFound As Integer

' Set up LTable2 and UTable2 (Lower and Upper-Bound Table 2s)

' The index value is the number of the child in the table.

' xTable2RIN is the RIN of the child.

' xTable2Coverage is the DNA coverage of the child, initially zero.

Dim UTable2RIN(1 To FPLimit) As Integer

Dim UTable2Coverage(1 To FPLimit) As Double

' If the person has DNA-tested, set value to 1 (100%).

' Then exit the function.

If RinatDNA(RinReceived) = True Then ' Do I need to write an outputrow for
this??

CalcUDNACoverage = 1

GoTo EndCalc ' Go to the exit of this function.

End If

' If you reach this point, the person has NOT done an autosomal DNA test.

' So, find Person X's children to calculate the person's coverage.

' To find the children, first find all the PersFam entries for which person X is a parent.

' That is, find all the families in which person X is a parent

' Search PersFam for the RIN of Person X (the row variable of PersFam).

' Then for each child of that MRIN (in FamPers), if the value (the RIN) is positive, Person X is a parent in
that MRIN.

' So, use that MRIN to search FamPers.

' For any FamPers record for this MRIN, if the second element (RIN) is negative, then that person is a
child of that parent in that family.

' So, capture the postive value of that child's RIN into Table 2 (both) as a child of Person X.

KidsFound = 0

For CountFamilies = 1 To PFLimit

If PersFam(RinReceived, CountFamilies) > 0 Then ' Have found a family record (MRIN) of Person X - so now find all children of that family and load to Table 2

MRINFound = PersFam(RinReceived, CountFamilies)

For CountChildren = 1 To FPLimit

If FamPers(MRINFound, CountChildren) < 0 Then

KidsFound = KidsFound + 1

UTable2RIN(KidsFound) = Abs(FamPers(MRINFound, CountChildren))

UTable2Coverage(KidsFound) = 0 ' Initialize the Child's DNA Coverage at zero

End If ' If FamPers(MRINFound, CountChildren) <> 0

Next ' For CountChildren = 1 To FPLimit

End If ' If PersFam(RinReceived, CountFamilies) <> 0

Next ' For CountFamilies = 1 To PFLimit

' So now, you have built both versions (L and U) of Table2.

' So calculate the key constants.

' N = number of children of the parent

' P = Pieces of the Venn diagram of the parent = $(2^N) - 1$

' M = Max percent of parent that each piece of the Venn diagram can contribute = $1/(2^N)$

' M is calculated as a decimal and not as a percent

Dim N, P As Long

Dim M As Double

N = KidsFound

P = $(2^N) - 1$

M = $1 / (2^N)$

' Now create both versions (L and U) of Table1. Each column is a separate 1-dim array.

' Each row of Table 1 has one column for each child with the binary value of that row number in those columns.

' Each row of Table 1 also has one column for M which is the constant M for all rows.

' Each row of Table 1 also has one column for W (weight)/

' -- For the lower bound W is the maximum coverage of any child with a 1 in their column in that row.

' -- For the upper bound W is the minimum of 1 or of the sum of the coverages of all the children with a 1 in their column in that row.

' Each row of Table 1 also has one column for R (result) which is $M * W$.


```
*****
*****
```

```
*****
```

```
' Set up LTable1 and UTable1 (Lower and Upper-Bound Table 1s)
' The index value is the number of the piece of the Venn diagram in the table.
' Thus the index number is the row number in Table1.
'
```

```
*****
```

```
Const Table1PieceLimit = 4096 ' Allows for up to 12 children
Const Table1KidLimit = 12 ' Allows for up to 12 children
```

```
Dim UTable1Binary(1 To Table1PieceLimit, 1 To Table1KidLimit) As Integer
Dim UTable1M(1 To Table1PieceLimit) As Double
Dim UTable1W(1 To Table1PieceLimit) As Double
Dim UTable1R(1 To Table1PieceLimit) As Double
```

```
*****
```

```
' Set the Binary Value for Each Piece/Row in the Child Columns
' Convert Venn Diagram piece number to its binary equivalent and put bits into Table1 child cells
' Works by reducing piece number by power of 2 on each iteration
```

```
*****
```

```
For PieceNum = 1 To P
    UTable1W(PieceNum) = 0
    UTable1M(PieceNum) = M
    UTable1R(PieceNum) = 0
    Remainder = PieceNum
    For KidNum = 1 To N
        BitPower = N - KidNum
        Bit = WorksheetFunction.Power(2, BitPower) ' Power(2, BitPower)
        If Remainder >= Bit Then
            UTable1Binary(PieceNum, KidNum) = 1
            Remainder = Remainder - Bit
        Else
            UTable1Binary(PieceNum, KidNum) = 0
        End If ' If Remainder >= Bit
    Next ' For KidNum = 1 To N
Next ' For PieceNum = 1 To P
```

```
*****
*****
```

```
' Now that both tables are set up, step through each child to obtain their DNA coverage in Table2.
' This is done be recursive calls to this same routine.
```

```
*****
*****
```

```
For ChildCount = 1 To N
    ChildRIN = UTable2RIN(ChildCount)
```

```

TargetUDNACoverage = CalcUDNACoverage(ChildRIN)
UTable2Coverage(ChildCount) = TargetUDNACoverage
Next ' For ChildCount = 1 To N

'+++++
+++++
'+++++ THE KEY CALCULATION
+++++
'+++++
+++++
' Now the children have all been found and their own DNA Coverage has been calculated.
' So, calculate the coverage of their parent.
' This is the KEY CALCULATION of this function.
' It is the only place that the calculation of the lower and upper bounds differ.
'+++++
+++++
'+++++
+++++
'+++++
+++++
WorkingCalcUDNACoverage = 0
For PieceCount = 1 To P
'
*****
*****
' Calculate upper bound W = max of the products of each child's coverage (from Table 2)
' Calculate upper bound R = W * M
'
*****
*****
    SumChildDNA = 0
    For ChildCount = 1 To N
    '
*****
*****
        ' Calculate lower bound W = max of the products of each child's coverage (from Table 2)
        '
*****
*****
            ChildProduct = UTable2Coverage(ChildCount) * UTable1Binary(PieceCount, ChildCount) * M
            SumChildDNA = SumChildDNA + ChildProduct
            UTable1W(PieceCount) = SumChildDNA
        Next 'For ChildCount = 1 To N
        UTable1R(PieceCount) = WorksheetFunction.Min(UTable1W(PieceCount), UTable1M(PieceCount))
        WorkingCalcUDNACoverage = WorkingCalcUDNACoverage + UTable1R(PieceCount)
    Next ' For PieceCount = 1 To P

```

```

'
*****
*****
' Set the output value to be returned
'
*****
*****
CalcUDNACoverage = WorkingCalcUDNACoverage

'
*****
*****
' Now that the calculation for this person is complete, write the output rows
' for this person and the children and Tables 1 and 2 values.
'
*****
*****
OutputValue =
"++++"
"

OutputRow(LineCount) = OutputValue
LineCount = LineCount + 1

OutputValue = "UPPER BOUND DNA Coverage of RIN " & RinReceived & " (" & RinName(RinReceived) &
") = " & CalcUDNACoverage
OutputRow(LineCount) = OutputValue
LineCount = LineCount + 1

OutputValue = "Children of RIN " & RinReceived & " (" & RinName(RinReceived) & ")"
OutputRow(LineCount) = OutputValue
LineCount = LineCount + 1

For ChildCount = 1 To N
    OutputValue = "RIN " & UTable2RIN(ChildCount) & " (" & RinName(UTable2RIN(ChildCount)) & ")"
    OutputRow(LineCount) = OutputValue
    LineCount = LineCount + 1
Next ' For ChildCount = 1 to N

'
*****
*****
' Write out Table 2
'
*****
*****

```

```

OutputValue = "***** Table 2 *****"
OutputRow(LineCount) = OutputValue
LineCount = LineCount + 1

OutputValueRIN = ""
OutputValueCoverage = ""
For ChildCount = 1 To N
    OutputValueRIN = OutputValueRIN & UTable2RIN(ChildCount) & "--"
    OutputValueCoverage = OutputValueCoverage & UTable2Coverage(ChildCount) & "--"
Next ' For ChildCount = 1 to N

OutputRow(LineCount) = "RINS--" & OutputValueRIN
LineCount = LineCount + 1

OutputRow(LineCount) = "DNA--" & OutputValueCoverage
LineCount = LineCount + 1

'
*****
*****
' Write out Table 1
'
*****
*****
OutputValue = "***** Table 1 *****"
OutputRow(LineCount) = OutputValue
LineCount = LineCount + 1

OutputValuePiece = ""
OutputValueHeader = "----R----M----W<<<<"
For ChildCount = 1 To N
    OutputValueHeader = OutputValueHeader & UTable2RIN(ChildCount) & "+"
Next ' For ChildCount = 1 To N

OutputRow(LineCount) = OutputValueHeader
LineCount = LineCount + 1

OutputValuePiece = ""
For PieceCount = 1 To P
    OutputValuePiece = OutputValuePiece & "----" & UTable1R(PieceCount) & "--" &
    UTable1M(PieceCount) & "--" & UTable1W(PieceCount) & "<<<<"
    For ChildCount = 1 To N
        OutputValuePiece = OutputValuePiece & UTable1Binary(PieceCount, ChildCount) & "+"
    Next ' For ChildCount = 1 to N
    OutputRow(LineCount) = OutputValuePiece
    LineCount = LineCount + 1
    OutputValuePiece = ""
Next ' For PieceCount = 1 to P

```

EndCalc:
End Function ' CalcLDNACoverage

Appendix C: Understanding the Visual Basic Program

While the program looks formidable, it really is quite simple. Unfortunately, the Visual Basic editor only allows for the addition of comment lines and also inserts underscore lines between sections but otherwise is primitive for providing understanding.

The program has these sections.

1. Declarations Section (the first section with no title)
2. Sub DNACoverage()
3. Function InitVars()
4. Function ReadGEDCOM()
5. Function CalcLDNACoverage(ReceivedRIN)
6. Function CalcUDNACoverage(ReceivedRIN)

Do not consider this an exemplar of ideal programming. It is a quick (as quick as the primitive Visual Basic editor and error messages allow) and a bit dirty program that could probably be cleaned up a lot. It is not at all state-of-the-art.

1-Declarations Section

The declarations section sets up all the “global” constants, variables and arrays. “Global” means that these objects can be used in any part of the program. A sub-function (such as the one that reads the GEDCOM file) can use “local” constants, variables and arrays that cannot be used in other sections of the program.

2-Sub DNACoverage()

Think of this section as the orchestra conductor. A simple line here can actually be an instruction to a sub-function than can lead to a great deal of computation. While there is a fair amount of housekeeping going on in the section for the eventual reporting of the results, it really does five basic things.

1. Initiate the variables that will be used to store the program’s internal representation of the GEDCOM file.
2. Read the GEDCOM file and store it into the internal representation.
3. Calculate the lower bound coverage of the target person.
4. Calculate the upper bound coverage of the target person.
5. Print the results report on the “Results Report” worksheet.

3-Function InitVars()

This brief section simply initializes the internal representation of the GEDCOM file as not knowing anything. This ensures that there are no faulty conclusions reached at any later decision point.

4-Function ReadGEDCOM()

This routine reads the GEDCOM and internally stores the information on the people and their relationships and which ones have DNA-tested.

5-Function CalcLDNACoverage(ReceivedRIN)

This is one of the two sections that are the real engines of the program: this one that calculates the lower bound and the other one that calculates the upper bound. These are the recursive sections which call more executions of themselves as the processing burrows down through the family tree and then back up to calculate the DNA coverage of a single parent from the coverage of all their children who have DNA coverage.

The two functions (lower and upper bound) are identical except for the step in which all the children's coverages are combined to calculate the parent's coverage.

The function receives the RIN of the person whose DNA coverage is to be calculated. It then finds all the children of that parent in the GEDCOM file (as internally represented). The function sets up the internal Table 1 representation of the Venn diagram for this parent and their children.

The function then steps through the children one by one. If a child has tested, their coverage is set at 100%, but if not then the function calls itself with the RIN of the child as the person whose DNA coverage is to be calculated.

6-Function CalcUDNACoverage(ReceivedRIN)

This is the other shoe in the pair. This function calculates the upper bound of the DNA coverage of the designated person from the coverage of his/her children. It is identical to the lower bound step except in how it calculates the parent's coverage once the coverage of all the children is known. See the main part of this paper for the specifics of how those two different coverages are calculated.

ⁱ *Journal of Genetic Genealogy*, vol. 10, no. 1, Fall 2022.

ⁱⁱ My real purpose is that all family tree software will someday have autosomal DNA coverage as a standard feature, just as we have relationship calculation and other tools.

ⁱⁱⁱ The Hurricane Path "Cone of Uncertainty" Analogy: The "cone of uncertainty" in hurricane path predictions provides a useful analogy. Different models forecast different paths that when all shown on the same map look like different strands of spaghetti noodles. The cone of uncertainty holds all those different predicted paths. In the same way, the range of DNA coverage holds all possible combinations that the descendant DNA tests can provide for the coverage of the target person. We cannot know in a theoretical model just which is the actual combination possible from those DNA tests. Once the raw DNA results for all the tests are combined, they will define a single combination. But with only the theoretical and not the actual results, we can only calculate a range of possible combinations.

^{iv} For cases where there are no generations for which there are no tested children, there is no range so that a single number can represent the DNA coverage. In such cases, the average and both the lower and upper bounds are all the same. This situation only exists in theoretical models. When actual DNA segments are known and can be included in the calculation, there is no longer uncertainty. Thus, the DNA reconstruction in GFG or in Borland Genetics will result in a specific percent of coverage based on the actual DNA results and not on a theoretical model that only provides estimates.

Simulations can also give a sense of reality. In hurricane forecasting, these are the "spaghetti" charts that align the hurricane paths predicted by different models. The different paths give a sense of the size of the cone of

uncertainty. In the same way, running many simulations of actual family trees give a strong sense of the size of the range of DNA coverage. The simulations of Amy Williams and the broader complex family tree simulations of Briton Nicholson explored this. Even for recent connections where the theoretical models can provide a single estimate, the simulations result in a range of possibilities with the theoretical model's calculated coverage usually right in the middle. The benefit of the theoretical model for ranges is that it can definitively calculate the upper and lower bounds so that the full possibility of the range of coverages can be known with certainty (within the limits covered in Appendix A).

^v In a lineage-linked relational database, these two operations (top-down tracing of the descendants and bottom-up propagation of the DNA coverage) work in parallel in the same call of the sub-routine. In the Graphs for Genealogists implementation, Dr. David Stumpf leveraged the power of the graph database to replace the top-down tracing of the descendants with database queries that created directed graphs connecting the target ancestor with each tested descendant.

^{vi} The only estimates that a recursive algorithm can accurately calculate are the lower and upper bound. Attempting to propagate any other number (such as the average) within the range inevitably leads to distortion, either toward over or under estimation. The distortion is minimal for fewer children but becomes significantly greater with more children if the children do not have high DNA coverage themselves.

^{vii} <https://dnainter.com/tools/sharedcmv4>

^{viii} Nicholson, Briton. "The Overused CentiMorgan". Dec. 31, 2020. (<https://dna-sci.com/2020/12/31/the-overused-centimorgan>)

^{ix} Nicholson, Briton. "Modeling the Inheritance of Ancestral DNA". Feb. 8, 2020. (<https://dna-sci.com/2020/02/08/modeling-the-inheritance-of-ancestral-dna>)

^x Borland, Kevin at <https://www.facebook.com/groups/borlandgenetics/posts/404094730131696>

^{xi} Stumpf, Dr. David at <https://www.wai.md/post/ancestor-reconstruction>