IN THE UNITED STATES DISTRICT COURT
FOR THE WESTERN DISTRICT OF PENNSYLVANIA

UNITED STATES OF AMERICA	)	
	)	
v.	)	Criminal No. 19-369
	)	
LAFON ELLIS	)	

DECLARATION OF DR. MATS HEIMDAHL AND DR. JEANNA MATTHEWS

We, Mats Heimdahl, Ph.D, and Jeanna Matthews, Ph.D., declare we have personal knowledge of the following, and if called upon to do so, could and would testify completely to the matters contained herein.

PRELIMINARY STATEMENT

1. TrueAllele is a black box. Information is entered, and out comes a result that could cause an otherwise innocent person to be convicted. The prosecution, by the directive of Dr. Mark Perlin who runs Cybergenetics, the company that produced TrueAllele, does not want Mr. Ellis' Defense team, this Court, or anyone else to look inside the box. Acceding to that request would ignore sound science.
2. TrueAllele is a probabilistic genotyping ("PG") software program that purports to conduct extraordinarily complex mathematical computations to analyze DNA samples that cannot be analyzed by traditional methods. Put differently, TrueAllele is attempting to solve a problem that *cannot be verified manually*, meaning that lab users cannot check the system's accuracy. The declaration of Dr. Perlin, provided to the defense, says "TrueAllele is used to analyze DNA evidence, particularly in cases where human review might be less reliable or not possible." Troublingly, no one outside of Cybergenetics, TrueAllele's developer, knows

how TrueAllele truly works. Software cannot be evaluated without full access to executable source code and related documentation. No one has been granted such access.

3. Even simple software programs are prone to flaws. A misplaced number, an incorrect assumption, or an unaccounted-for limitation can result in a failure. Problematically, such flaws are often latent and go undetected. The opportunities for and consequences of such flaws increase dramatically for complex software programs such as TrueAllele. Recent history is littered with examples of seemingly small flaws causing catastrophic failures.

4. It is virtually certain that there are flaws in the TrueAllele software. On average, there will be six flaws for every 1,000 lines of code, and TrueAllele has 170,000 lines of code. Given its nature, TrueAllele is particularly likely to contain undetected flaws: users are unlikely to notice failures, the incentive structure makes reporting flaws less likely, and the TrueAllele software has not been subject to thorough, independent review.

5. Flaws have been discovered in other PG programs—including STRmix and Forensic Statistical Tool ("FST")—and in much simpler technologies such as breathalyzers. Those flaws—which called into question thousands of convictions—frequently went undiscovered until the source code was reviewed as part of the judicial process. Mr. Ellis' request to subject TrueAllele to such scrutiny is not only prudent but essential to determining whether TrueAllele operates as Cybergenetics claims.

QUALIFICATIONS OF THE AUTHORS

6. The authors of this declaration are experts in engineering, testing, and validating computer systems, including forensic software. We were hired by Mr. Ellis' attorney, Khasha Attaran, to submit this declaration to explain why it is essential that Mr. Ellis has full access to the

TrueAllele source code.

7. Dr. Mats Heimdahl is a Distinguished University Teaching Professor and the Head of the Department of Computer Science & Engineering at the University of Minnesota. Before his appointment as Head of Computer Science & Engineering, he served as the Director of the University of Minnesota Software Engineering Center for eleven years and Director of Graduate Studies for the Master of Science in Software Engineering program. Professor Heimdahl earned his PhD at the University of California, Irvine in Information and Computer Science, and has received numerous awards for his research in high-assurance software development, for example, a National Science Foundation CAREER Award, a NASA Turning Goals Into Reality Award, and the NASA Office of Safety and Mission Assurance Best Overall Research Project Award.

8. Professor Heimdahl publishes widely on the engineering of safety critical software systems, software requirements, software testing, and software verification and validation in top peer-reviewed journals and conferences where his publications have received numerous best paper citations. He has testified to the National Academies of Sciences, Engineering, and Medicine on "Software Certification and Dependability" and served on the National Academies of Sciences, Engineering, and Medicine's committees on the "Review of the Enterprise Architecture, Software Development Approach, and Safety and Human Factor Design of the Next Generation Air Transportation System" and "Airport Passenger Screening: Backscatter X-Ray Machines". He has also served as an expert in numerous cases involving software engineering.

9. Dr. Jeanna Matthews is a Full Professor of Computer Science at Clarkson University and an affiliate at Data & Society. She has a Ph.D. in Computer Science from the University of

California Berkeley, one of the top computer science departments in the world. She is a member of the Association for Computing Machinery ("ACM") Council, founding co-chair of the ACM Technology Policy Subcommittee on Artificial Intelligence and Algorithm Accountability, and a member of the ACM Technology Policy Committee. She has particular expertise in evaluating the role of advanced technology in criminal justice. In 2017 and 2018, Dr. Matthews was a principal investigator on a multidisciplinary project funded by the Brown Institute, a collaborative institute centered at Columbia University and Stanford University, to focus specifically on the analysis and comparison of probabilistic genotyping software. Dr. Matthews has authored or co-authored dozens of publications, including multiple articles focused on PG and has written and spoken widely on the need for increased algorithmic accountability and transparency, especially in criminal justice software.

WE DECLARE

I. Software Faults Are Ubiquitous

10. Source code is the human-readable formal plan for software that provides the instructions for how the computer will function. Simple programs can require thousands of lines of code, and complex programs, such as TrueAllele, can require hundreds of thousands or millions of lines of code. Source code faults are ubiquitous and difficult to detect. As software complexity increases, so does the risk of faults.

A. The nature of software and software development

11. Software engineering entails three primary "domains" essential to provide a software solution; the problem to be solved and its context needs to be understood, a solution to that problem needs to be devised, and a software system implementing that solution needs to be

constructed. An error in any of these domains may lead to flaws in the resultant software system. The "Simple Example" below will serve as an illustration of the challenges of software engineering. Problem Identification involves thoroughly understanding the problem to be solved. In the simple example below, the problem to be solved is adding the numbers from 1 through *n* and reporting the sum. The problem to be solved must be thoroughly understood so we actually solve the right problem (what if the client really wanted a software system that computed the *product* as opposed to the *sum*?) and we must thoroughly document the problem (if not documented, how can we ever ensure that the software being produced actually solves the problem?) For complex problems, thoroughly understanding and documenting the problem is a known issue and exceedingly difficult.

Simple Example: the three domains of software engineering		
Problem Identification	**Computational Solution Development**	**Software Construction**
The need to compute the sum of the first *n* whole numbers (*i.e.*, 1+2+3+4+…+*n*) based on a user inputting the value of *n*.	Option 1: add each number until the first *n* numbers are added. Option 2: compute *n**(*n*+1)/2. Both algorithms arrive at the same result: 1+2+3+4=10 4*(4+1)/2=10	An engineer creates a program to implement the algorithm, requiring two inputs from the user: the number *n* and the first name of the user making the request. The program also gets the date and time from the computer on which it is running. The program outputs the name, date and time of the request,n, and the sum.

12. To solve the problem, we derive a computational solution. In our simple example, we consider two solutions. Option 1 is straightforward and it is easy to determine that if we build software that does this, we have solved our problem. Option 2 is far more efficient, but it is also harder to argue that it actually solves the problem at hand (in this case it can be demonstrated through a mathematical proof). In short, the proposed solution must be thoroughly validated to determine that it solves the problem; this validation can take many

complementary forms (e.g., proof, inspection, peer review of the code, testing using a software prototype, testing using the production software, etc.).

13. The implementation of the computational solution is achieved through the construction of software and this step can and does introduce substantial new possibilities for errors to be introduced. One might think that after the computational solution has been developed and validated, the implementation of the software is a "trivial" matter, but this is patently false. When implementing software, even if our computational solution is absolutely correct, there are countless ways in which software can fail to perform as anticipated. For example, the software might work for users named "Bob," but give the wrong result for users named "Mohamed," because the software was inadvertently designed to handle only names of five characters or fewer. More troublingly, the software might print the wrong sum for certain numbers, giving the correct answer when n is 92,672, but the wrong answer when n is 92,689 because the highest number that the program can process is 4,294,967,295 (due to a numeric overflow since the implementation used a "data type" that could only store numbers up to a certain size).

14. These types of failures occur, in part, because software is non-continuous and does not behave like traditional engineered systems. Traditional engineered systems follow the laws of physics, which makes them relatively straightforward. Consider a simple crane used to lift boats at the local marina. The crane has been designed to handle x kilograms before failing. Once the crane has been shown to lift boats weighing x kilograms, one can be confident that it will lift boats weighing 0.4 x kilograms. The result would not differ depending upon the day of the week, the color of the boat, or the name of the boat's owner. The same cannot be said for the software used to process the handling of the boats. If the

software has not been programmed to handle certain cases (such as weekend transactions, red boats, or long owner names), it might fail entirely or produce erroneous results that could go undetected.

15. Because software is non-continuous, test results cannot be interpolated; in theory, any input could cause a failure. One cannot assume that because a software program gives the correct result when a user inputs 3,000 or 5,000 the software will work when the user enters 4,000. Because the number of potential inputs to a program is astronomical, testing them all is impossible, and defense experts should have access to step through the execution of TrueAllele source code on the exact inputs provided in Mr. Ellis' case. This would require access to the source code and the ability to step through its execution line by line on the input provided in Mr. Ellis' case.

16. Since the software might fail entirely for any given input, or, worse, produce erroneous results that appear plausible, validation studies such as the ones cited as evidence that TrueAllele is reliable and trustworthy are utterly inadequate to demonstrate the correctness of the TrueAllele software. The TrueAllele studies are important steps towards establishing that the statistical models underlying TrueAllele are adequate (such studies do help validate the computational solutions); however, they reveal exceedingly little about the quality of the software implementation of that computational solutions and cannot rule out errors that could impact the result for any given set of inputs and in particular for the set of inputs provided in Mr. Ellis' case.

B. Validation and verification of software

17. The two terms validation and verification (V&V) are often confused and used interchangeably. In software development, validation generally refers to attempting to

answer the questions "Did we build the right system?" whereas verification attempts to answer "Did we build the system right?" If we consider our simple system in the example, validation might involve running a few numbers *n* through the software, checking that the result really is the sum of the numbers 1 through *n* (validate that we got the computational solution right), and ask the client if this looks right to them (validate that we got the problem description right). Verification might involve carefully selecting test cases to determine if the software has any issues with numeric accuracy, if it properly rejects negative numbers, if it works with short and long user names, if it works in all timezones and times of day, if it works with very large numbers, etc., etc., these tests are carefully constructed to determine if the software implementing the computational solution has been built correctly, not to test if the computational solution is correct.

18. In the case of TrueAllele, the validation of the computational solution has taken the form of various published validation studies where a small number of test cases using varying DNA mixtures are analyzed; these studies are all designed to test (to validate) the computational solution, there is absolutely no mention of ensuring that the software will operate correctly in all circumstance (there is no attempt to verify that the software always accurately implements the computational solution). Notably, in these published studies those conducting the testing typically have incentives to show the software is working well, rather than independent, third-party testers who would be more likely to have incentives to identify cases where the software is failing. A defense expert working for Mr. Ellis would have incentives to specifically examine the operation of the software on the exact inputs in Mr. Ellis' case and identify problems. No one involved in any of the various published validation studies would have been tasked with this work or had this incentive.

19. Furthermore, even if we, for the sake of argument assume that the computational solution solves the problem, there is no evidence or discussion in these publications about whether or not the TrueAllele software faithfully implemented this solution. The fundamental misunderstanding between validating the computational solution in general, and verifying that a specific software implementation faithfully implements that computational solution has permeated the reasoning of the government in this case and is also sadly a common problem with how the law treats software results more broadly. We believe this is why so many courts have regularly accepted software results without sufficient validation and verification evidence, but fortunately this is starting to change (*e.g.* Judge Neff's decision in *United States v. Gissantaner*, 417 F. Supp. 3d 857, 864 (W.D. Mich. 2019) (excluding STRmix™ after allowing defense access to the source code for inspection). More simply stated, the over 35 validation studies that Cybergenetics cites to establish its software's reliability and that prosecutors tout as evidence of its reliability, speak only to the general applicability of the computational solution proposed and say absolutely nothing about the reliability of the software implementation itself.

20. The published and peer reviewed studies do provide some evidence that the computational solution is viable. They do not, however, provide any evidence that the TrueAllele software is reliable, correct, and will always produce the results advertised. The evidence of correct implementation of the TrueAllele software resides in the internal documentation of Cybergenetics, in the requirements specifications, design descriptions, version control history, test plans, test cases, problem or issue tracking, inspection reports, the traceability from requirements to tests, traceability from tests to code, test adequacy measurement reports, the source code itself, etc.; artifacts that have not been provided to the courts or

defense for examination. Without a careful examination of these artifacts, there is simply no way of determining if the TrueAllele software is working as advertised in all cases.

21. As an analogy, let us say there is a new and novel DNA analysis technique, this technique would obviously need to be validated to determine whether or not it works. The persons who invented this technique might validate it in their lab, they may publish the procedure and results, and a few others may perform replication studies by sending their data to the same lab for analysis and confirm that it seems to work. This is, in essence, what the published studies of TrueAllele do; they describe the data being delivered to TrueAllele and they evaluate the results coming back. However, this cannot rule out many classes of errors that could be introduced when the technique is implemented in software or errors in software that could be triggered by specific inputs in specific cases such as Mr. Ellis'.

22. Continuing our hypothetical example, if the proposed technique was used in a criminal case and the results came out of the lab indicating a very high likelihood ratio, the defense could rightly want to ask about the way in which the proposed technique was actually implemented in the lab. The defense might ask to inspect the lab to determine that it is not contaminated and that the machines are properly calibrated, to talk to the persons performing the analysis to see if they have the proper training and background, to determine that they are following other best practices etc. If in this situation the owner of the lab denied the defense this access and even failed to provide other evidence of internal procedures and controls such as regular reports auditing their internal procedures, this would not be acceptable. The lab would not be able to claim simply that "the methodology that underlies the results has general acceptance in the relevant scientific community"; they must demonstrate that the methodology has been implemented correctly and acceptably, and that

the established procedures were followed. In our analogy, the TrueAllele software itself is the "lab", if we cannot inspect it, how it was constructed, the internal procedures and controls, etc., we have no idea what analysis was actually performed on this particular sample, if the software is accurate, free from defects, etc.; an obviously unacceptable situation. Cybergenetics is asking the defense to just trust that the initial general published studies guarantee that they are always producing the correct results. The evidence provided by Cybergenetics simply does not address the many possible sources of error that can affect the TrueAllele software.

23. In other industries producing life critical software, for example, complex medical devices or air-transport avionics software, there are strict guidelines for both validation and verification of the software systems. As examples, the development of complex medical devices requires a Quality System per section 520(f) of the Federal Food, Drug, and Cosmetic Act. Such a quality system necessitates rigorous documentation and design of such devices to demonstrate that the devices are safe and effective. Similarly, development of airborne software for air-transport aircraft must follow "RTCA/DO-178C, Software Considerations in Airborne Systems and Equipment Certification" that requires, among many other things, evidence that the software has been tested to exercise the requirements, that all lines of source code have been executed by some test, that robustness test have been performed (e.g., tests for numeric and memory related problems), etc. Legal requirements such as section 520(f) of the Federal Food, Drug, and Cosmetic Act, and standards such as RTCA/DO-178C and IEEE 1012-2012, require extensive documentation, traceability, and carefully constructed tests precisely because treating the software as a black-box during

validation and verification is wholly inadequate[1]. Given that TrueAllele is used to potentially make life or death decisions, we should demand similar levels of rigor in the software development. Cybergenetics should be ready and able to produce this documentation and evidence on demand to demonstrate to the defense and the court that their software, indeed, operates as advertised.

24. The prosecution in this case writes, "In terms of expert disclosures, it is difficult to imagine a more comprehensive disclosure than what Cybergenetics has provided this defendant." *See* Doc. No. 108 at 13. We do not even need to go so far as to imagine a more comprehensive disclosure. We can look to the types of V&V materials produced for life critical software, for example, complex medical devices or air-transport avionics software. When a defendant's life and liberty is on the line, criminal justice software is life critical software, a disclosure with the source code and other V&V materials as requested in the defense's discovery motion and subpoena to Cybergenetics is necessary.

C. Opportunities for error are higher in complex programs

25. Most software flaws result from simple mistakes. The more complex the program, the greater the risk of flaws. "A research study has shown that professional programmers average six software defects for every 1000 lines of code (LOC) written." Hoang Pham, *Software Reliability*, *in* WILEY ENCYCLOPEDIA OF ELECTRICAL AND ELECTRONICS ENGINEERING 565, 565 (John G. Webster ed., 1999).

26. In many situations, software faults are not discovered until they result in obvious and catastrophic failures. For example, NASA's Mars Climate Orbiter exploded because the software controlling its thrust was written to use English units instead of metric units. *See*

[1] R. W. Butler and G. B. Finelli, "The infeasibility of quantifying the reliability of life-critical real-time software," in IEEE Transactions on Software Engineering, vol. 19, no. 1, pp. 3-12, Jan. 1993, doi: 10.1109/32.210303.

Andrew Pollack, *Missing What Didn't Add Up, NASA Subtracted an Orbiter*, N.Y. Times (Oct. 1, 1999).[2] Two Boeing 737 MAX aircraft crashed because a software modification made the aircraft vulnerable to nosedives. *See* Niraj Chokshi, *House Report Condemns Boeing and F.A.A. in 737 Max Disasters*, N.Y. Times (Sept. 16, 2020).[3] An Ariane 5 rocket exploded because a fault in a program tried to "stuff a 64-bit number into a 16-bit space." James Gleick, *Little Bug, Big Bang*, N.Y. Times (Dec. 1, 1996).[4] A "software glitch" in Therac-25 radiation therapy machines resulted in cancer patients receiving excessive radiation when a certain group of commands was entered. *Fatal Radiation Dose in Therapy Attributed to Computer Mistake*, N.Y. Times (June 21, 1986).[5]

27. Software can also have errors that cause subtle yet serious failures. For example, software faults caused a jury selection program to exclude zip codes where most African-Americans lived. Nina W. Chernoff, *No Records, No Right: Discovery & the Fair Cross-Section Guarantee*, 101 Iowa L. Rev. 1719, 1723-24, 1731-32 (2016). As the Sixth Circuit recognized, "[t]he glitch was a mistyped parameter in the software, buried in a mountain of computer code, that was only discovered after a broad statistical analysis led to an extensive internal investigation." *Ambrose v. Booker*, 684 F.3d 638, 645 (6th Cir. 2012).[6] Such latent faults are the hardest errors to find and fix, leading to undetected harm over many years.[7]

---

[2] https://nyti.ms/34GaQCR.
[3] https://nyti.ms/2GL0E3u.
[4] https://nyti.ms/2GRp0sA.
[5] https://nyti.ms/34wZpx6.
[6] Another example of a latent fault is a software bug that mistakenly caused convicts in Washington to be released early from prison for more than a decade. See Michelle Shephard, *More Than 3,200 US Prisoners Have Been Released Early Because of a Software Glitch*, BBC News (Dec. 23, 2015), https://bbc.in/2FfsDIh.
[7] While most software faults are unintentional, complex source code also provides bad actors with an opportunity to do harm in subtle ways. A high-profile example is Volkswagen's use of buried source code on 11 million cars to cheat state emissions tests. Mike Spector & Mike Colias, *Volkswagen Pleads Guilty to Criminal Charges in Emissions-Cheating Scandal*, Wall St.

II. TrueAllele's Source Code Likely Contains Undetected Flaws

28. Numerous factors suggest that TrueAllele is likely to contain undetected flaws, including that: (1) flaws have been discovered in other PG programs and less complex forensic tools, often only after source code was produced pursuant to judicial orders; (2) flaws are unlikely to be noticed because forensic lab users cannot check the system's accuracy; (3) unlike commercial software, the incentive structure for forensic software does not encourage reporting flaws; (4) TrueAllele has not been subject to thorough, independent review; and (5) experts have raised concerns about TrueAllele's lack of reproducibility.

A. Other forensic programs, including PG programs, have been found to contain faults

29. Recent history is littered with examples of latent flaws in forensic software being discovered after the software was used in numerous arrests and convictions. These examples include not only flaws in other PG software programs, but also flaws in much simpler programs.

30. In total, at least thirteen "coding faults" have been found in STRmix, TrueAllele's chief competitor.[8] In one notable example, the miscode impacted 60 criminal cases, requiring new likelihood ratios to be issued in 24 cases. David Murray, *Queensland Authorities Confirm 'Miscode' Affects DNA Evidence in Criminal Cases*, Courier-Mail (Mar. 20, 2015).[9]

31. Another PG program, FST, avoided independent review for years until a federal judge ordered source code disclosure. Stephanie J. Lacambra et al., *Opening the Black Box: Defendants' Rights to Confront Forensic Software*, The Champion, May 2018, 28, 32. The defendant's expert discovered that "[a] secret function . . . was present in the software, tending to overestimate the likelihood of guilt," and that "[t]he actual functioning of the

---

J., (Mar. 10, 2017), https://on.wsj.com/30MqxXR.
[8] *See* STRmix, *Summary of Miscodes* https://bit.ly/36ILKWi (last updated Sept. 15, 2020).
[9] https://bit.ly/34DBlZy.

software, revealed upon inspection of the source code, did not use the methodology publicly described in sworn testimony and peer-reviewed publications." *Id.*[10] Once these flaws were discovered, a high-profile conviction based on FST analysis was overturned. *See* Alan Feuer, *Hasidic Man Convicted of Beating Black Student Gets Verdict Overturned*, N.Y. Times (Oct. 10, 2018).[11]

32. In recent years, thousands of faults have been discovered in the source code of top breathalyzer systems. According to a 2019 *New York Times* investigation, breathalyzers "generate skewed results with alarming frequency," and "Judges in Massachusetts and New Jersey have thrown out more than 30,000 breath tests in the past 12 months alone." Stacey Cowley & Jessica Silver-Greenberg, *These Machines Can Put You in Jail. Don't Trust Them*, N.Y. Times (Nov. 3, 2019).[12] After the New Jersey Supreme Court granted source code access, defense experts found that the code was "littered with 'thousands of programming errors.'" *Id.* Similar errors were found across the country. *Id.*

33. It is unlikely that TrueAllele is somehow free of flaws. Rather—as was the case with the forensic tools discussed above—TrueAllele's flaws will simply go undetected until defendants are granted meaningful access to the source code and other software development artifacts developed in the validation and verification of the software.[13]

B. Flaws in TrueAllele are unlikely to be noticed

---

[10] *See also* Jeanna Neefe Matthews et al., *When Trusted Black Boxes Don't Agree: Incentivizing Iterative Improvement and Accountability in Critical Software Systems*, 2020 Proc. AAAI/ACM Conf. on AI, Ethics, & Soc'y 102, 103 (FST developers "aggressively resisted expert witness review that could have exposed the problem for 5 years while using the output of the system as evidence in over 1000 serious criminal cases.").

[11] https://nyti.ms/33GHuoD.

[12] https://nyti.ms/3jHAyNt.

[13] *See* Natalie Ram, *Innovating Criminal Justice*, 112 Nw. U. L. Rev. 659, 682 (2018) ("In the few cases in which courts have compelled disclosure of private source code . . . reviewers identified significant errors in almost every instance.").

34. Although some faults may generate an error message or crash a program entirely, other faults operate more stealthily. A fault in TrueAllele's source code would likely not prevent the system from generating a match statistic; it would just prevent the system from generating an *accurate* match statistic. And in the vast majority of cases, the technician operating the machine would never be able to tell that the result was incorrect.

35. Such dormant flaws are a bigger problem for complex programs such as TrueAllele. In simpler systems, the user can evaluate the program's results based on other forms of analysis. For example, if a computer's calculator application indicates that two plus two equals five, the user can check the math by performing the addition mentally. Not so with TrueAllele.

36. With PG, the software is attempting to solve a problem that *cannot be verified manually*. Technicians cannot "check the math" on the results, because it is not humanly possible to perform TrueAllele's calculations. The only indication of what the right answer "should be" is the result from TrueAllele. Even if a match statistic was inflated by a factor of a million, the lab would probably not be able to tell that a failure had occurred.

37. It is one thing to match a high-quality evidence sample (e.g., lots of blood from one person) to a suspect. It is another thing to attempt to match small samples from multiple people (e.g., DNA from fingerprints on a gun that multiple people have touched). Whereas methods for manually interpreting high-quality DNA samples from individual people (termed "high-template, single-source" samples) have been fairly standard for decades, low-template, mixed samples are much more difficult and cannot be manually verified leaving analysts with little choice but to accept the output of software like TrueAllele as the truth without question.

C. Cybergenetics and law enforcement have incentives to not identify or report software flaws

38. In case after case, including Mr. Ellis', software companies and governments have demonstrated a determination to avoid subjecting PG programs to meaningful scrutiny. This is likely due, in part, to the unique incentive structure that applies to forensic software. In traditional commercial software, failures are often discovered and reported by customers. The manufacturer has a financial incentive to quickly correct the flaws. With criminal justice software, however, this incentive structure does not necessarily exist. The customers are prosecutors, but those harmed by software flaws are criminal defendants. *See* Matthews et al., supra note 10, at 103.[14]

39. Even if state employees were able to detect failures in TrueAllele's source code, they may not have an incentive to report those flaws. TrueAllele allows prosecutors to achieve convictions where they otherwise would not. If the results support conviction, there is a risk that the government will not reliably report failures.[15] Similarly, Cybergenetics may not have an incentive to identify flaws in TrueAllele either. A flaw's discovery could cast doubt on the product's reliability, undermine prior convictions, and threaten Cybergenetics' financial future.

40. This incentive structure likely impacted how the government handled the case of Florencio Jose Dominguez. Due largely to results from a PG program, Dominguez was convicted of murder and sentenced to 50 years to life. Greg Moran, *Murder Case that Highlighted DNA-*

[14] There are incentives to dismiss defendants' claims that results in their cases could not be accurate. *See* Jeanna Matthews et al., *You're Just Complaining Because You're Guilty: A DEF CON Guide to Adversarial Testing of Software Used in the Criminal Justice System* (Aug. 11, 2018), https://youtu.be/4cscBvDYP-Q.

[15] For example, significant lapses and misconduct at the FBI Crime Lab went unreported and/or uninvestigated for years. *See* David Johnston, *Report Criticizes Scientific Testing at F.B.I. Crime Lab*, N.Y. Times (Apr. 16, 1997) (quotation marks omitted), https://nyti.ms/3nPWWXc.

Analysis Controversy Ends with Plea to Reduced Charge, Release, San Diego Union Trib. (Dec. 6, 2019).[16] When Dominguez's counsel moved to reopen the case based on suspicions about the DNA analysis, a California judge ordered the disclosure of the program's source code. *Id.* But instead of giving Dominguez a chance to conduct an evaluation of the software—during which flaws could be discovered—the prosecutors allowed him to walk free. *Id.* The government's actions suggest that it would go to great lengths to prevent the discovery of flaws in the software.[17]

D. Existing testing of TrueAllele is incomplete and unreliable

41. The prosecution in this case writes that TrueAllele has been subject to "Over thirty-five validation studies . . . to establish the reliability of the TrueAllele method and software. Eight of these studies have been published in peer-reviewed scientific journals, for both laboratory-generated and casework DNA samples." *See* Doc. No. 108 at 5.

42. We examined each of the validation studies provided by the prosecution in preparation for this declaration (a total of 39 documents including 9 documents classified as peer reviewed articles). We focused on the 9 peer reviewed articles and concluded the following: 1) The validation studies were not independent, 2) The validation studies did nothing to address the quality of the implementation or likelihood of implementation errors and 3) The validation studies were incomplete. We also point out that peer review of validation studies is simply not the same as software validation and verification. Peer review in a scientific journal indicates only that the reviewers thought the scientific community should see the results, not

[16] https://bit.ly/3nszGOZ.

[17] *See* Matthews et al., *supra* note 10, at 102 ("[D]evelopers may be tempted to avoid costly debugging by claiming intellectual property protection in order to keep knowledge of known problems away from defendants . . . ."); Lacambra et al., *supra*, at 38 ("[P]rosecutors consistently urge courts to . . . deprive criminal defendants of access to forensic software.").

that the software is reliable enough to be used in the criminal justice system in general or in any specific case in particular.

1. The validation studies were not independent

43. One serious flaw in the TrueAllele "validation" work cited by the prosecution is that those studies originated almost exclusively from within Cybergenetics' orbit. Of the over 35 validation studies cited by the prosecution the majority came from Cybergenetics, Dr. Perlin (who runs Cybergenetics), or law enforcement agencies. Id. Perlin himself is an author of 7 of the 9 documents classified as peer reviewed articles and in another the authors explicitly acknowledged the "helpful comments and guidance provided by Dr. Mark Perlin, Matthew Legler, and William Allan of Cybergenetics." The last peer reviewed paper explicitly states that "This paper does not address the application of the TrueAllele expert review system to the analysis of forensic samples."

44. The lack of independent review raises serious concerns about the reliability of the studies, and was the chief criticism of PG software, including TrueAllele, in a report by the President's Council of Advisors on Science and Technology (the "PCAST Report").[18] Specifically, the PCAST Report cautions that a method is not foundationally valid unless it has "be[en] shown, based on empirical studies, to be *repeatable, reproducible*, and *accurate*, at levels that have been measured and are appropriate to the intended application." *Id.* at 4. Additionally, it indicates that validity as-applied places "special emphasis on ensuring that the method was applied correctly and within its empirically established range." *Id.* at 82. The PCAST Report further calls for more testing that "should be performed by or should include independent research groups not connected with the developers of the

[18] President's Council of Advisors on Sci. & Tech., Exec. Office of the President, *Forensic Science in Criminal Courts: Ensuring Scientific Validity of Feature-Comparison Method* 78-81 (Sept. 2016), https://bit.ly/34D6L1X.

methods and with no stake in the outcome." *Id.* at 81.[19]

2. The validation studies did nothing to address the quality of the implementation or likelihood of implementation errors

45. None of the validation studies cited by the prosecution in this case are addressing the problem of assessing the overall correctness of the implementation of the TrueAllele source code. Dr Perlin's own declaration states "Source code is not used in validation studies."

46. Thus, even if the studies support the soundness of the TrueAllele algorithm (or the computational solution to the DNA analysis problem as we referred to it in our Simple Example above), it is impossible for them to establish that the algorithm is correctly implemented in the TrueAllele software (or the Software Construction as we referred to it in our Simple Example above). Without access to the program source code as well as all supporting software development artifacts, researchers can say no more than that the results generated by the program are plausible for the few test cases that were executed. But, the software could appear to produce plausible results while still concealing latent errors.

47. Quite simply, studies that are designed to validate the computational solution cannot be used to verify that the program is operating as expected in all circumstances; to make such a determination a testing regime designed specifically to exercise all corners of the software must be designed (e.g., along the guidelines outlined in DO-178C) and used in conjunction with a thorough inspection of all necessary development artifacts expected in the development of a life critical system, including the source code:

> *[R]eliance on validation studies in place of source code access, rather than alongside it, is likely insufficient to verify that software has performed as its designer claims. In part, this stems from the limited verification that can be gleaned from "black-box testing"—testing that "considers only the inputs and outputs of a system or component." As*

[19] A cornerstone of the gate-keeping test for expert opinions in civil cases is independent validation and reproducibility. *See Daubert v. Merrell Dow Pharms., Inc.*, 509 U.S. 579, 593 (1993). Surely, a lesser standard should not apply in criminal cases.

> *technologists have explained, "[c]omputer scientists . . . have shown that black-box evaluation of systems is the least powerful of a set of available methods for understanding and verifying system behavior." More powerful and effective is "white-box testing," in which "the person doing a test can see the system's code and uses that knowledge to more effectively search for bugs." Accordingly, researchers have concluded that, to enable effective scientific inquiry, "anything less than the release of source programs is intolerable . . . ."*

Ram, *supra*, at 688-89 (citations omitted).

3. The validation studies were incomplete

48. Another major shortcoming "of all the published TrueAllele® validation studies is that the number of samples tested was relatively small." William C. Thompson et al., *Forensic DNA Statistics: Still Controversial in Some Cases*, The Champion, Dec. 2012, 12, 20.[20] Because of the non-continuous nature of software, the results from a small set of inputs cannot be reliably interpolated into cases involving different sets of inputs. Unless the DNA profiles and contribution proportions analyzed in this case are similar to the limited samples used in the validation studies, those studies are of little value here.

49. Experts who have evaluated the validation studies have doubts about applying those studies to complex cases:

> *[E]xisting validation is insufficient to prove that TrueAllele® can consistently make correct genotype inferences in challenging, problematic cases such as mixture cases with unequal contributions from the contributors, limited quantities of DNA, degradation due to environmental insult, etc.*

Id.

50. Experts have criticized TrueAllele for its inability to reproduce results. Reproducibility "is a central requirement of the scientific process." Peter Ivie & Douglas Thain, *Reproducibility in Scientific Computing*, 51 ACM Comput. Surv. 3, 63:1 (July 2018). However, TrueAllele

[20] PG software is generally not validated for samples involving a large number (e.g., four or more) contributors, but, in practice, PG *is* used on such samples or where a large number of contributors cannot be ruled out (e.g., Touch DNA on a gun).

regularly produces dissimilar likelihood ratios from multiple analyses of the same sample. Thompson et al., *supra*, at 20. For example, in one case, Cybergenetics analyzed a single sample four times and produced four different likelihood ratios: 389 million, 1.9 billion, 6.03 billion, and 17.8 billion. *Id.* Mark Perlin chose to report the 6.03 billion number, reasoning that "it was the center of the range of values." *Id.* Such a degree of unexplained variation is troubling.

4. Peer review of validation studies is not the same as software validation and verification

51. Although some of the TrueAllele validation studies were published in peer-reviewed journals, it is important not to overstate what that means. A journal's decision to publish a study is based merely on whether the results should be seen by the scientific community; the job of a journal reviewer is simply to ensure the quality and integrity of scientific and scholarly research. Publication of a study designed to validate the computational solution implemented in software is not—and is not intended to be—a stamp of approval endorsing the program's use in the criminal justice system, nor an assurance that the software is correct, reliable, and free from defects--as discussed earlier, there is simply no information in the published studies to allow such assurance. Dr. Perlin's own declaration says "Over thirty-five validation studies have been conducted by Cybergenetics and other groups to establish the reliability of the TrueAllele method and software. Eight of these studies have been published in peer-reviewed scientific journals, for both laboratory-generated and casework DNA samples. Source code was not needed or used in any of these studies." Peer review in a scientific journal only indicates that the reviewers thought the scientific community should see the results, not that the software is reliable enough to be used in the criminal justice system in general or in any specific case in particular.

III. Full Access to the TrueAllele Source Code and Supporting Materials Is Necessary

52. Review of TrueAllele's source code is necessary to identify the existence and import of any flaws in the program. Review of the software must be meaningful, and cannot consist of merely reviewing the public materials Cybergenetics provided to the Defense. Without actually reviewing the verification and validation "V&V" materials, requested by Nathan Adams in his declaration, Mr. Ellis' review of the materials publicly disclosed would render expert review essentially meaningless. That is, TrueAllele's code cannot be meaningfully reviewed without full access to the executable source code and software development documentation. *See* Lydia Pallas Loren & Andy Johnson-Laird, *Computer Software-Related Litigation: Discovery and the Overly-Protective Order*, 6 Fed. Cts. L. Rev. 1, 14-18 (2012).

53. Three of the restrictions, in particular, render meaningful review impossible: (1) the inability to compile and execute the source code; (2) the inability to access software development documentation; and (3) the inability to communicate with subject-matter experts.

A. Access to executable software compiled directly from the source code offered during expert witness review and the ability to examine its operation in the specific inputs in Mr. Ellis' case is necessary

54. Preventing defense experts from analyzing, compiling, and executing the code makes it impossible for the experts to test the software or understand how it operates. *See id.* at 47 ("A clause permitting only handwritten notes is burdensome in the extreme."); *id.* at 53-54 ("[T]he prohibition on actually compiling the source code is mystifying."). Without the ability to actually execute the source code, Mr. Ellis' expert would be restricted to mentally executing the approximated 170,000 lines of code in TrueAllele, which is simply not a reasonable task to do manually. As computer scientists, we can state clearly that this is an extremely inefficient and burdensome way to look for problems in software.

55. Quite simply, other than by running the program, Mr. Ellis' expert cannot evaluate how the software actually operates, whether the results yielded by TrueAllele are reproducible, or whether it works the way that Cybergenetics claims. *See* Lacambra et al., *supra*, at 29. As experts in the field, we know that a complex system cannot be debugged using a pen and paper. Any legitimate search for flaws requires, at a minimum, running the program and observing how the system responds to various inputs. A meticulous paper review of source code might spot some obvious faults, but, in general, an expert would not be able tell the difference between an error and an unusual coding choice without actually running the software. *Id.* It is also impossible for an expert to know for sure that the source code they are given matches exactly the version used to evaluate the evidence in the case.

B. Access to supporting documentation is necessary

56. In order to evaluate the reliability of TrueAllele, the Defense's expert would need access to TrueAllele's software development documentation, including testing, software design, bug reporting, change logs, and program requirements. *See* Loren & Johnson-Laird, *supra*, at 17-18.

57. Such documentation not only acts as a "road map" for the expert to understand the source code, *id.* at 17, but also allows the expert to determine whether Cybergenetics followed industry standards in developing TrueAllele. Because software is error prone, there are industry standards for software verification and validation. *See, e.g.*, *IEEE Standard for System and Software Verification and Validation*, IEEE Std 1012-2012; Sci. Working Grp. on DNA Analysis Methods, *Guidelines for the Validation of Probabilistic Genotyping Systems* (June 2015).[21] Access to TrueAllele's software development documentation would

[21] https://bit.ly/3lrI13D.

allow an expert to determine whether those standards were followed.[22]

58. The software development documentation would also allow the expert to determine whether Cybergenetics followed the heightened protocols that should be applied to safety-critical systems, which are systems in which failures could cause loss of life, significant property damage, or environmental damage. We believe that TrueAllele is a safety-critical system, because a malfunction can result in the wrongful execution or incarceration of an innocent individual. An international standard known as IEC 61508 addresses the standards applicable to the development of safety-critical software systems.[23]

59. Access to development documentation also allows a reviewer to focus on areas where material faults might exist. *See* Loren & Johnson-Laird, *supra*, at 17-18. For example, such access would allow an expert to evaluate why Cybergenetics revised TrueAllele's source code more than twenty-five times. *See* Andrea Roth, *Trial by Machine*, 104 Geo. L.J. 1245, 1273 (2016). "[W]ith no published documentation as to what has been revised or why," *id.*, and without access to the source code, "it is impossible to know whether those changes corrected undisclosed errors or inadvertently introduced new ones." Ram, *supra*, at 681.

CONCLUSION

60. Mr. Ellis' very freedom hinges upon the results yielded by a black-box software program. Mr. Ellis and this Honorable Court deserve to understand how that software actually works,

[22] For example, such documentation would show if TrueAllele was audited through independent verification and validation (IV&V). IV&V is "[v]erification and validation (V&V) performed by an organization that is technically, managerially, and financially independent..." Ron Ross et al., Nat'l Inst. Of Standards & Tech., Special Pub. 800-160, *Systems Security Engineering: Considerations for a Multidisciplinary Approach in the Engineering of Trustworthy Secure Systems* 168 (2016).

[23] Systems are categorized by "safety integrity level," ranging from SIL1 to SIL4. Each safety level has additional requirements, with SIL4 being the most demanding. *See Functional Safety of Electrical/Electronic/Programmable Electronic Safety-Related Systems*, IEC 61508, 2010.

and the only means of doing so is by providing full access to the executable source code and the validation and verification materials. We disagree with the government's statement in the motion to quash subpoena that "[t]here is no genuine controversy as to the validity and reliability of the TrueAllele method." *see* Doc. No. 108 at 4. No one outside of Cybergenetics has examined the source code or validation and verification materials for TrueAllele. None of the over 35 studies provided looked at the source code or any of the TrueAllele validation and verification artifacts and are therefore unable to comment on how the general computational method is implemented in software. This is a crucial step in which many errors can be made and in which many errors have been made in similar software projects. Given its nature, TrueAllele is particularly likely to contain undetected flaws: users are unlikely to notice failures, the incentive structure makes reporting flaws less likely, and the TrueAllele software has not been subject to thorough, independent review. We ask the Court to recognize the real possibility of error in the complex software implementation of TrueAllele and allow the Defense team in this case to examine how that software processes the exact inputs at issue in Mr. Ellis' case before determining that the results of that system are fundamentally reliable without question.

Dr. Mats Heimdahl, Distinguished University Teaching Professor and Head of the Department of Computer Science & Engineering, University of Minnesota

D. Jeanna Matthews, Professor, Department of Computer Science, Clarkson University