CASE STUDY

Developing *STACK* practice questions for the Mathematics Masters Programme at the Open University

Grahame Erskine, School of Mathematics & Statistics, The Open University, Milton Keynes, UK. Email: Grahame.Erskine@open.ac.uk
Ben Mestel, School of Mathematics & Statistics, The Open University, Milton Keynes, UK. Email: Ben.Mestel@open.ac.uk

Abstract

The design and implementation of Masters level e-assessment in *STACK* is described for the introductory module *M820 Calculus of Variations and Advanced Calculus* in the Open University's Masters Programme in Mathematics. Some basic design principles are described and illustrated for an online practice quiz on the use of the Jacobi equation to classify stationary paths.

Keywords: e-assessment, *STACK*, design, implementation.

1. Introduction

Over the past 15 years there has been an explosion in the use of computer-based/online assessment in university mathematics, with several systems pioneered by UK practitioners. In addition to commercial systems such as *Maple TA* (Maplesoft.com, 2018), we have seen, for example, *Maths E.G.* at Brunel (Mathcentre.ac.uk, 2018), *Dewis* at UWE (Dewisprod.uwe.ac.uk, 2018), *Numbas* at Newcastle (Numbas.org.uk, 2018), and *STACK* (Stack.ed.ac.uk, 2018), originally at the University of Birmingham, but now adopted by several institutions, including The Open University (OU), where it has been demonstrated that *STACK* can work well at large scales[1] (Heacademy.ac.uk, 2018). See Figure 1 for the implementation of *STACK* at the OU.

Despite the theoretical basis, success, and growing adoption of these e-assessment systems in Mathematics (Sangwin, 2013), their use has largely been confined to undergraduate mathematics, typically at levels 1 and 2. Anecdotally, at least, there has been a feeling that these systems would be less useful for higher-level mathematics and that they are ill equipped to deal with mathematical argument and proof, especially given the wide variety of student responses and approaches. Such worries are understandable and the jury is still out on whether e-assessment can be effectively implemented at level 3 and above.

However, all mathematics involves some degree of calculation and students benefit from practice in the necessary techniques, even if such calculation is supplemented by nuanced argument, especially at higher levels. So, when we decided to implement practice questions for the Open University's MSc module *M820 Calculus of Variations and Advanced Calculus,* we were confident that we could devise questions that would be of genuine help to students. We also knew several formative e-assessment questions had already been successfully implemented by our colleagues on the module *M823 Analytic Number Theory 1,* the other entry module to the MSc programme. Indeed, if they had developed *STACK* questions for pure mathematics, then surely it would be straightforward to do so in the more conventional calculational world of calculus of variations.

Hence, armed with a grant from the university's programme to aid student retention, we set out to develop six significant *STACK* exercises to support students on M820 to develop the standard techniques in these key areas of the theory:

1. Solution of the Euler-Lagrange differential equation for quadratic functionals to obtain the stationary path.
2. Solution of variational problems through the first-integral, for those functionals permitting such an approach.
3. Local classification of stationary paths of functionals into minima, maxima and saddle points, through the analysis of the Jacobi differential equation.
4. Calculation of the Noether invariants (first-integrals) for functionals invariant under a scale change in the variables.
5. Diagonalisation of quadratic functionals involving two dependent variables, thereby allowing the stationary paths to be calculated by the solution of two independent variational problems.
6. The use of the Rayleigh-Ritz method to find an upper bound for the least eigenvalue of a Sturm-Liouville problem.

Each of these problems involves fundamental techniques that might be included in the end-of-module examination, on which assessment for the module is based. In this article we focus on the third technique, on the Jacobi equation.

2. Authoring online practice quizzes

Our first task was to set the overall design of the e-assessment questions. In this we were fortunate that the OU has an extensive history of e-assessment in mathematics, from computer-marked 'objective testing' (i.e. multiple choice), in which the university was a pioneer in the 1970s/1980s; through an elegant but now largely superseded Java-based system OpenMark (now open-source but non-maintained) (Open.ac.uk, 2018), (Butcher, 2008); to widespread implementation of *STACK* as part of an enhanced *Moodle* Quiz Engine (Hunt, 2012). Since both of us were familiar with *STACK* and had authored *STACK* questions in the past (albeit at Levels 1 and 2), the choice of system was clear. Since the individual topics involved multiple stages of calculations, we opted for each topic to be a separate *Moodle* quiz, with each calculation stage a separate question, each question linked so that they had the same instance of each random parameter[2]. This was done so that students could benefit from feedback on early parts of the calculation before tackling the later, harder parts.

We adopted the following *modus operandi* for developing the *STACK* questions.

1. Ben (as M820 Module Team Chair[3]) scoped out the possible questions which he felt were appropriate for STACK implementation, taking into account the importance of the material and the prospect for tractable, randomised questions which illustrated the theory and provided multiple practice examples without involving the student in unnecessary algebraic and arithmetic complexity.

2. After an initial discussion, Grahame (as former M820 student and professional programmer) then took the outline questions and implemented them in *STACK*, ironing out the multitude of niggles and problems.

3. Ben then checked and commented on the implementation, suggesting amendments, usually on wording.

4. A final discussion finalised the implementation before moving on to the next topic.

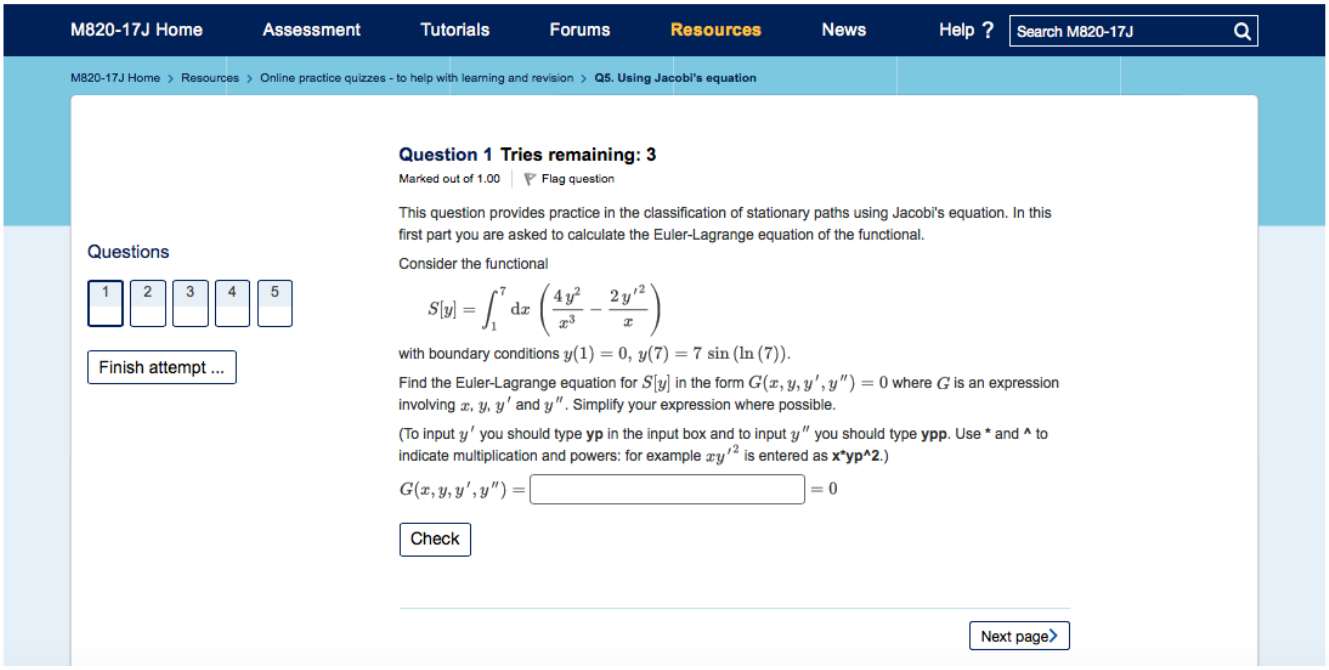


Figure 21. Implementation of *STACK* in the OU's *Moodle*-based Virtual Learning Environment. The example shown is the first part of a five-part practice quiz on the use of the Jacobi equation in the calculus of variations.

3. The calculus of variations and the Jacobi equation

Recall that the calculus of variations, dating from the 17th Century, studies extremal paths of functionals of the form $\int_a^b dx\, F(x, y, y')$ (and its many generalisations) together with assorted boundary conditions and constraints.

The candidate extremals are typically given by stationary paths, which are solutions of the Euler-Lagrange equation $\frac{d}{dx}\left(\frac{\partial F}{\partial y'}\right) - \frac{\partial F}{\partial y} = 0$. The theory, with its generalisations to higher dimensions and higher derivatives, has applications in many fields including physics (where it forms the basis of many modern physical theories), biology, control engineering, economics and chemistry. The modern theory incorporates singular and infinite functionals, but the focus of the M820 module is on hands-on calculation rather than on abstraction. See Gel'fand *et al.* (2000) and MacCluer (2012) for readable introductions, two of a plethora of excellent textbooks on this classical area of mathematics.

To provide the background for our case study, we now outline the use of the Jacobi equation to classify stationary paths.

4. Example: the Jacobi equation

Once the stationary paths have been calculated, the next step is to classify them into local maxima, local minima and saddles. Unfortunately, this is often a difficult task. One approach, a generalisation of the second derivative classification of stationary points of real functions, is to solve the Jacobi differential equation initial value problem

$$\frac{d}{dx}\left(P \frac{du}{dx} \right) - Q\, u = 0, \qquad u(a) = 0,\ u'(a) = 1,$$

where $P = \frac{\partial^2 F}{\partial y'^2}$, and $Q = \frac{\partial^2 F}{\partial y^2} - \frac{d}{dx}\left(\frac{\partial^2 F}{\partial y \partial y'}\right)$, evaluated on the stationary path $y(x)$. If $u(x)$ has no zeros ("conjugate points" in the jargon) in the half-open interval $(a, b]$ then $y(x)$ is a local minimum if $P > 0$ on $[a, b]$; a local maximum if $P < 0$ on $[a, b]$; and a saddle if P changes sign on $[a, b]$. Further analysis in needed in other cases. Unfortunately, despite being a linear equation, the Jacobi equation is frequently difficult to solve analytically. Therefore finding clean, tractable examples for students to practise the method is a challenge for educators.

Our aim for an online practice exercise on the Jacobi equation was to present the students with a functional, necessarily specialised in form, and to lead them through the solution of the Euler-Lagrange equation to find the stationary path and to then classify the stationary path using Jacobi's equation.

In order to reduce the calculations, we opted to work with a quadratic functional of the form

$$\int_a^b dx\, (\alpha_0\, x^m y'^2 + \beta_0\, x^{m-2} y^2)\,,$$

defined on the closed interval $[a, b]$, with fixed end-point boundary conditions $y(a) = A,\ \ y(b) = B$, where A, B are constants, chosen to reduce the algebraic complexity of the solution. The constants α_0, β_0 and m were also specialised as described below.

This type of functional has two distinct advantages. First, for functionals quadratic in y' and y, the Euler-Lagrange equation and the Jacobi equation turn out to be the same equation, with the important difference that for the Euler-Lagrange equation there are boundary conditions at the two endpoints a and b, while the Jacobi equation is an initial value problem.

Second, for this type of functional, the Jacobi equation is a second-order Euler differential equation which can be readily solved by converting to a homogeneous constant-coefficient differential equation. Specifically, the Jacobi equation is

$$A_2 x^2 \frac{d^2u}{dx^2} + A_1 x \frac{du}{dx} + A_0 u = 0, \quad u(a) = 0, \qquad u'(a) = 1,$$

where $A_2 = \alpha_0,\ A_1 = m\alpha_0$, and $A_0 = -\beta_0$. The usual transformation $x = e^t$, leads to

$$A_2 \frac{d^2u}{dt^2} + (A_1 - A_2) \frac{du}{dt} + A_0 u = 0, \quad u(\log a) = 0, \qquad u'(\log a) = 1.$$

Since our principal aim is to provide practice in the use of the Jacobi equation to classify stationary paths, we look for choices for which the classification varies with the solution of this initial value problem. Thus we choose the auxiliary equation to reduce to the form $\lambda^2 - 2\rho\lambda + \rho^2 + \omega^2 = 0$, where ρ and ω are 'nice' constants with $\omega \geq 0$ and ρ and ω not both zero. The solution (in terms of x) is $u(x) = \left(\frac{a}{\omega}\right)\left(\frac{x}{a}\right)^\rho \sin(\omega \log(x/a))$ for $\omega > 0$ and $u(x) = a\left(\frac{x}{a}\right)^\rho \log(x/a)$ for $\omega = 0$.

At this point several things are apparent. First, to avoid singular behaviour, we must take $0 < a < b$. Second, although $\omega = 0$ is an important special case from a pedagogical perspective, its inclusion would complicate the implementation, so that it would be better to restrict to $\omega > 0$ in the first instance, leaving the case $\omega = 0$ for future development.[4] Third, the solution for $a \neq 1$ is significantly more complex so that a restriction to $a = 1$ would simplify computations for the students, at the risk of not providing practice in the more complex cases. Compromises such as these are frequent in e-assesment and the trick is to balance the pedagogy and the programming.

So, restricting to $a = 1$ and $\omega > 0$, gives $u(x) = \frac{1}{\omega} x^{\rho} \sin(\omega \log x)$ and there are zeros of $u(x)$ in $(1, b]$ if and only if $\omega \log b \geq \pi$. By varying the choice of ω and b, we are able to flip between the two cases: there are no zeros and the Jacobi test may be applied, and there is at least one zero in $(1, b]$ and the Jacobi test fails. In the first case, the stationary path is a minimum if $P(x) = 2\alpha_0 > 0$ and is a maximum if $P(x) = 2\alpha_0 < 0$. Note that for this functional P is of fixed sign (for $\alpha_0 \neq 0$) so the stationary path is either a local maximum or a local minimum.

It follows that we can randomise the problem by making the following choices:

a. ω, a random small positive integer, e.g., in the range 1..3
b. m, a random integer, e.g., in the range -1..5 excluding 0
c. α_0, a random integer, e.g., in the range -3..3 excluding 0
d. b, a random integer greater than 1, e.g., in the range 2..9

and setting $A_2 = \alpha_0$, $A_1 = m\,\alpha_0$, $A_0 = \alpha_0 \frac{(m-1)^2 + 4\,\omega^2}{4}$, $\rho = \frac{1-m}{2}$, $\beta_0 = -A_0$.

However, these randomisations may not lead to a desired distribution of the three cases: minimum, maximum or test failure (conjugate points), and so it was necessary to restrict b further depending on the other parameters, randomising to some extent the choice of outcome, to ensure that the successful cases occurred sufficiently often. Certainly, development is an iterative process.

Now, the Euler-Lagrange equation is

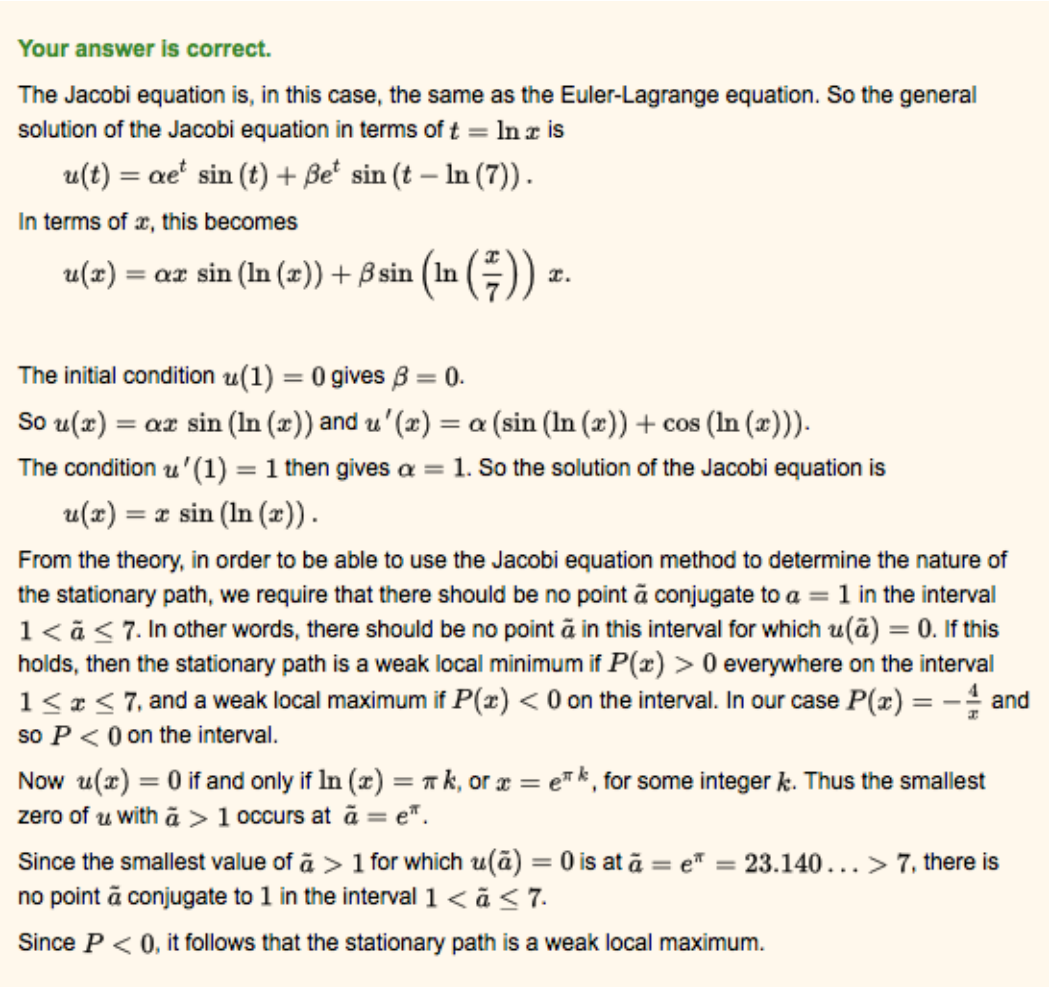
$$A_2 x^2 \frac{d^2y}{dx^2} + A_1 x \frac{dy}{dx} + A_0 y = 0, \qquad y(a) = A, \qquad y(b) = B,$$

Choosing, for simplicity, $A = 0$ and then setting $B = kb^{\rho} \sin \omega \log b$, where k is a small random positive integer, gives a simple formula for the stationary path $y(x) = k\, x^{\rho} \sin(\omega \log x)$.

In our implementation for the students, the problem was split into five sequential questions with the same instance of the randomisation (with extensive feedback, as shown in Figure 2):

1. Calculate the Euler-Lagrange equation;
2. Transform the resulting Euler equation using the transformation $x = e^t$;
3. Solve the resulting constant-coefficient differential equation and substitute back to get the stationary path;
4. Calculate $P(x)$ and $Q(x)$ and hence the Jacobi equation;
5. Solve the Jacobi initial value problem to find $u(x)$, investigate the existence of conjugate points and hence, if possible, classify the stationary path.

A great strength of STACK is the ability to tailor feedback to student input. At this development stage, our use of this facility was confined to providing hints for incorrect tries and to providing feedback if an expression is not in the right variables, if a correct answer can be simplified, if the appropriate differential equation or boundary/initial value is not satisfied, and if the final classification of the stationary path is incorrect.



Your answer is correct.

The Jacobi equation is, in this case, the same as the Euler-Lagrange equation. So the general solution of the Jacobi equation in terms of $t = \ln x$ is

$$u(t) = \alpha e^t \sin(t) + \beta e^t \sin(t - \ln(7)).$$

In terms of x, this becomes

$$u(x) = \alpha x \sin(\ln(x)) + \beta \sin\left(\ln\left(\frac{x}{7}\right)\right) x.$$

The initial condition $u(1) = 0$ gives $\beta = 0$.

So $u(x) = \alpha x \sin(\ln(x))$ and $u'(x) = \alpha(\sin(\ln(x)) + \cos(\ln(x)))$.

The condition $u'(1) = 1$ then gives $\alpha = 1$. So the solution of the Jacobi equation is

$$u(x) = x \sin(\ln(x)).$$

From the theory, in order to be able to use the Jacobi equation method to determine the nature of the stationary path, we require that there should be no point $\tilde{a}$ conjugate to $a = 1$ in the interval $1 < \tilde{a} \leq 7$. In other words, there should be no point $\tilde{a}$ in this interval for which $u(\tilde{a}) = 0$. If this holds, then the stationary path is a weak local minimum if $P(x) > 0$ everywhere on the interval $1 \leq x \leq 7$, and a weak local maximum if $P(x) < 0$ on the interval. In our case $P(x) = -\frac{4}{x}$ and so $P < 0$ on the interval.

Now $u(x) = 0$ if and only if $\ln(x) = \pi k$, or $x = e^{\pi k}$, for some integer k. Thus the smallest zero of u with $\tilde{a} > 1$ occurs at $\tilde{a} = e^{\pi}$.

Since the smallest value of $\tilde{a} > 1$ for which $u(\tilde{a}) = 0$ is at $\tilde{a} = e^{\pi} = 23.140\ldots > 7$, there is no point $\tilde{a}$ conjugate to 1 in the interval $1 < \tilde{a} \leq 7$.

Since $P < 0$, it follows that the stationary path is a weak local maximum.

Figure 2. Feedback giving a worked solution for a correct answer to the final part of the Jacobi equation quiz. Although not shown here, *STACK* facilitates feedback tailored to a student's individual input.

5. Our experience of authoring questions at MSc level

In this section we outline our experiences of authoring in *STACK* at the MSc level in a series of vignettes.

1. **Two developers working together.** Having a fairly complete outline of each question before embarking on the development was helpful. One benefit is that two people worked through the details of the questions, one at the pen-and-paper specification stage and the other at the *STACK* implementation. Given the complexity of the material, this much reduced the probability of major errors in the question logic. As it turned out, most of the amendments made in the later stages were presentational – clarifying wording and so on.

   One thing to be aware of when tackling questions of this complexity is that multi-part questions, where students are led through the question in well-defined stages, bring added work to the development process to keep all the parts *in sync*. So estimation is key – when the question outline is 20 handwritten pages it is likely that the development effort required is substantial.

2. **Avoiding arithmetically and algebraically complex questions.** One aim was to develop 'clean' questions which were not overly algebraically and arithmetically complex. From the experience of setting exam questions over the years, there was a belief that it was important to start with clean answers and to work backwards to generate clean questions. In previous projects, attempts

have been made to keep the answers numerically reasonable, since students may be put off by the need to input answers involving very large numbers, or odd-looking fractions, square roots and so on. In this case the problem was compounded by the multi-part nature of most of the questions.

It turned out that on a couple of the questions the best strategy was to pick "nice" values for intermediate answers rather than the final answer. Working backwards then gave reasonable values for the question parameters, and then the final answers were not too awkward because the intermediate values had been chosen to be clean.

In designing the quizzes, we were conscious both of the need for tractable examples with clean arithmetic/algebra so that students focus on the higher-level skills and for realistic examples of sufficient complexity to give students practice in a range of situations. Certain of our choices leaned heavily towards arithmetic/algebraic simplicity (leaving more complicated cases for future development), but we were also mindful of the need for sufficiently many randomised parameters to span a large enough slice of the space of possible questions. Ultimately, it must be a matter of judgement. Perhaps, in retrospect, restricting to $a = 1$ was too limiting, although a wide variety of questions was nonetheless available to the students.

3. **Skill set needed to develop *STACK* questions.** When developing questions like these one is drawing on skills and knowledge from previous experience. Having some background in software development is helpful, but almost more important is the mathematical intuition. As a former student of this module, Grahame was able to think from the student's perspective about the logical steps required to develop the solution.

   It is more or less essential when developing these types of question that you should understand the material in enough detail to be able to work through the solution. Just as important is the ability to think about what can go wrong. What might confuse a student and how can I word the question or solution to be as helpful as possible? What errors are likely and can these be detected to give more helpful feedback than a generic "incorrect answer, try again" response?

4. **The most challenging problem during implementation.** These questions were at a level of complexity beyond anything we'd done before, so the usual minor issues were magnified. Often the *Maxima* system underlying *STACK* will decide on how to order the terms in a computed expression, and you have limited control over that. So sometimes you're fighting the system a little and need specific *LaTeX* for formatting. The nature of the subject is that complex expressions involving partial derivatives and subscripted variables are common, so the *LaTeX* formatting job is non-trivial.

   Probably the most awkward part was to come up with the input values which led to reasonable answers. In addition to picking good values for intermediate results, sometimes the only way to do that was the brute-force method of running all possible question values in a reasonable range through a model of the question logic built in *Maxima*, and selecting those parameters which result in sensible results.

5. **Advice to others contemplating writing e-assessment using the *STACK* system.** The benefits of *STACK* for this kind of mathematics, especially involving algebraic manipulation and calculus, are considerable. There is good material available for learning the basics of *STACK*, aimed primarily at developing questions in the early undergraduate range. Gaining experience of the system on questions at this level would be sensible before embarking on developing the more advanced questions in our project.

One thing that worked very well in our case, and which we recommend, is the detailed outline of the questions that we had on paper before embarking on the actual build on the system. If you try to develop the question on the fly, then things will go wrong, especially since many of our questions had four or five linked parts. Planning is key!

6. Conclusion and some general principles

The six online quizzes were completed in a six-week period in December 2017 – January 2018, each taking about 4 person-days from design to implementation, including checking. The M820 students were offered the six practice quizzes as a tool for learning and for examination revision and many used them for those purposes. Each quiz has been accessed over 220 times and by at least 85% of the 93 students taking the examination, but a detailed analysis of their effectiveness, and of their reception by the students, will be made once the module results are known.

The focus in this article has been on the design and implementation phase and on our experience at higher-level mathematics. Certainly, we have found the creation of these online quizzes to have been both enjoyable and intellectually stimulating, albeit with a hefty dose of frustration when the system (usually *Maxima*) did not work entirely as expected.

We end with some pointers to those who are looking to design their own e-assessment material, whether for formative or summative assessment. These are naturally pretty high level and most have already been embedded by mathematics and statistics educators as part of their normal day-to-day practice.

1. Start small and work up. It's best not to start with the most general case, but to deal with a basic case and add complexity afterwards, possibly as separate questions.
2. Avoid complicated answers. Use small integers/rationals where possible and avoid complicated intermediate algebra and arithmetic, the bane of computer algebra systems generally.
3. Specify the solution and work backwards to problem, or, possibly, work backwards and forwards from an intermediate step. Moreover, whenever you have an equation to solve (linear, differential, algebraic) specify the solutions first and then derive the equation, not vice versa, and be prepared to re-think your randomisation choices during development if the solutions are too complex.
4. Check the students' answers directly, don't try pattern matching with a model solution. The variety of student answers far outstrips our own imagination!
5. Often the hardest part is formatting the solutions.
6. Select randomly from a set of good parameter choices, rather than randomise each parameter independently.

7. References

Butcher, P., 2008. Online assessment at the Open University using open source software: Moodle, OpenMark and more. In: Khandia, F. (ed.). *Proceedings of the 12th CAA International Computer Assisted Assessment Conference*, 8th and 9th July 2008. Lougborough University, UK. pp. 65-78. Available at: https://research.moodle.net/149/1/Butcher_P_final_formatted_n1_2.pdf [Accessed 12 June 2018].

Dewisprod.uwe.ac.uk., 2018. *The Dewis e-Assessment System*. Available at: http://dewisprod.uwe.ac.uk/ [Accessed 6 June 2018].

Gel'fand, I. M., Fomin, S. V. and Silverman R. A., 2000. *Calculus of variations*. Mineola, N.Y.: Dover Publications.

Heacademy.ac.uk., 2018. *The Open University - STACK | Higher Education Academy.* Collaborative award for teaching excellence. Available at: https://www.heacademy.ac.uk/person/open-university-stack [Accessed 6 June 2018].

Hunt, T., 2012. *Computer-marked assessment in Moodle: Past, Present and Future.* In: D. Whitelock et al. (eds). Proceedings of the 2012 CAA International Computer Assisted Assessment Conference, 10th – 11th July 2012, University of Southampton.

MacCluer, C. R., 2012. *Calculus of variations: mechanics, control and other applications.* Mineola, NY: Dover Publications.

Maplesoft.com., 2018. *Maple T.A. - Online Assessment System for STEM Courses - Maplesoft*. [online] Available at: https://www.maplesoft.com/products/Mapleta/ [Accessed 6 June 2018].

Mathcentre.ac.uk., 2018*. Maths E.G.* Available at: http://www.mathcentre.ac.uk:8081/mathseg/ [Accessed 6 June 2018].

Numbas.org.uk., 2018. *Really versatile maths e-assessment | Numbas.* Available at: https://www.numbas.org.uk [Accessed 6 June 2018].

Open.ac.uk., 2018. *OpenMark Examples*. Available at: http://www.open.ac.uk/openmarkexamples/ [Accessed 12 June 2018].

Sangwin, C., 2013. *Computer Aided Assessment of Mathematics.* Oxford: Oxford University Press.

Stack.ed.ac.uk., 2018. *STACK*. [online] Available at: http://www.stack.ed.ac.uk [Accessed 6 June 2018].

[1] *STACK* services over a million e-assessment questions annually at the OU.
[2] The ability to serve multipart problems with consistent randomisations between the parts is a powerful feature of the integration of *STACK* within the *Moodle* Quiz.
[3] The Module Team Chair is the person in overall charge of the module and who leads the team of colleagues (which includes the OU's famed Tutors) who deliver the module to the students.
[4] Alternatives and special cases are often best developed as separate *STACK* questions, leaving the random selection of question type to *Moodle*.