Gas Field Multi-period Optimization Under Harmonic Decline

Chayut Wongkamthong
cw403@duke.edu

1 Summary

In operating a gas field, petroleum firms would like to maximize the condensate production. This process is usually done with a single-period linear optimization. However, some production constraints, especially the Harmonic decline pattern of the wells, are nonlinear and nonconvex. Moreover, the solution to a single-period optimization problem can be drastically different from the multi-period one as the former ignores the time domain completely.

In this project, we formulate a multi-period production optimization problem with Harmonic decline as a nonlinear optimization problem. We then solve it with the augmented Lagrangian technique and use the gradient projection method with trust-region to solve the subproblem. We show that the technique works perfectly under different scenarios by reporting the corresponding production schemes and explaining their semantics.

2 Background and theory

In section 2.1, we introduce the problem of gas field optimization over multiple periods. In section 2.2, we describe our method of optimization called the augmented Lagrangian technique. We delay the topic of numerical implementation to section 3.1.

2.1 Gas field optimization under Harmonic decline

In operating a gas field, petroleum firm needs to produce natural gas at the nomination rate to satisfy the gas sale agreement (GSA) made with the buyer whenever possible. Meanwhile, they try to maximize condensate production to increase their revenue. To achieve this, they collect fluid properties from different gas wells and construct production decline curves to capture the production potential over time (Wongkamthong, Wongpattananukul, Suranetinai, Vongsinudom, & Ekkawong, 2018). A specific type of such curve is called Harmonic decline where the production rate at time t is described by $q(t) = q_{pot}/(1 + dt)$ where q_{pot} is the initial potential of that well and d is the decline rate (Arps, 1945). The average production rate q_g over time period Δt can be calculated by integrating the aforementioned rate equation from the open time $t^{(j-1)}$ to the close time $t^{(j)}$ of any period j of that well. This leads to the relation $q_g \times \Delta t = \frac{q_{pot}}{d} ln(\frac{1+dt^{(j)}}{1+dt^{(j-1)}})$. Note that $t^{(j)} - t^{(j-1)}$ needs not be Δt as we might not open that well for the full period. This equation poses a nonlinear and, specifically, non-convex constraint in gas field optimization. Attempts have been made to solve nonlinear problems including using Taylor's series approximation to nonlinear constraints (Lo, Starley, & Holden, 1995) or simplifying it by doing single-period optimization, which ignore the time-dependent factors of field development. As this constraint is still smooth, we would like to apply numerical optimization technique to try to obtain a local solution that could represent a good field production plan. We pose our problem as follows where T denotes the number of production timesteps to be considered and N denotes the number of production wells. We consistently use the index i for well indexing and j for timestep indexing.

$$max_{t_i^{(j)}, q_{g,i}^{(j)}} \sum_{j=1}^{T} \sum_{i=1}^{N} CGR_i \times q_{g,i}^{(j)} \times \Delta t^{(j)} \times Price^{(j)} \tag{1}$$

$$s.t \quad q_{g,i}^{(j)} \times \Delta t^{(j)} = \frac{q_{pot,i}}{d_i} ln(\frac{1 + d_i t_i^{(j)}}{1 + d_i t_i^{(j-1)}}) \quad \forall j \in \{1, \ldots, T\}, \forall i \in \{1, \ldots, N\} \tag{2}$$

$$0 \leq t_i^{(j)} - t_i^{(j-1)} \leq \Delta t^{(j)} \quad \forall j \in \{1, \ldots, T\}, \forall i \in \{1, \ldots, N\} \tag{3}$$

$$\sum_{i=1}^{N} q_{g,i}^{(j)} \times \Delta t^{(j)} \leq Q_{nom}^{(j)} \quad \forall j \in \{1, \ldots, T\} \tag{4}$$

Instead of performing single-period optimization, which can be considered as a greedy process, the objective here is to maximize profit from selling condensate over the field life. In this project, we assume that the condensate gas ratio (CGR) of each well is fixed which is practical. The price model has a depreciation factor of 0.9 indicating that the monetary value in the following timestep is 90% of the previous one. Equation 2 represents the harmonic decline constraint for each well at each timestep. Equation 3 limits the possible production time at each step. We use Δt of 1 throughout our simulation. The last set of constraints is to satisfy maximum nomination rate Q_{nom} which we assume to be a fixed constant of 30 unit.

2.2 Augmented Lagrangian (AL) approach

The augmented Lagrangian (AL) method differs from the quadratic penalty method in the sense that, in AL approach, we introduce explicit approximation of the Lagrange multipliers associated with each of the constraints and update them as we iterate through the optimization process. This helps solve the issue of ill conditioning when the penalty parameter in the quadratic penalty method is high.

Concretely, we first handle inequality constraints that are not simple bound constraints by introducing slack variables. Thereby, the typical problem to be solved will be of the form $minf(x)$ such that $c(x) = 0$ and $l \leq x \leq u$ where we gather all optimization variables, including slack variables, into a vector x and all constraints into a vector-valued function $c(x)$. We then form the augmented Lagrangian function $\mathcal{L}_A(x, \lambda; \mu) = f(x) - \lambda^T c(x) + \frac{\mu}{2} c(x)^T c(x)$ where λ is the current approximation of the Lagrange multipliers and μ is the constant representing penalty for infeasibility.

At iteration k, with the current value of λ^k and μ_k, the subproblem to be solved is $min_x \mathcal{L}_A(x, \lambda^k; \mu_k)$ such that $l \leq x \leq u$. It can be shown that the first order optimality condition of this subproblem is $x = P(x - \nabla_x \mathcal{L}_A(x, \lambda^k; \mu_k), l, u)$ where $P(x, l, u)$ is the projection function of each element of x onto the box constraint from l to u. After finding an approximate solution x_k to this subproblem, we then proceed to test whether the current iterate x_k is feasible enough. If that is the case, we then update our estimated Lagrange multipliers using the formula $\lambda^{k+1} = \lambda^k - \mu_k c(x_k)$ while keeping the penalty unchanged $\mu_{k+1} = \mu_k$. Otherwise, we do not update λ^{k+1} but increase the penalty to force the next iterate to move to the region where it is more feasible. It can be shown that the iterate x_k can get closer to the local solution of the original problem x^* as we either increase μ_k or obtain a better approximation of the Lagrange multipliers λ^k.

There are several methods in solving the subproblem of the AL process. For example, in the nonlinear gradient projection method with trust-region, we solve the subproblem by using trust-region method. For each iteration in the trust-region submodule, we approximate the objective function using a quadratic approximation centered at the current iterate. Formally, we try to solve $min_d \frac{1}{2} d^T[\nabla^2_{xx} f(x_k) + \mu_k A^T A - \sum_i(\lambda_i^k - \mu_k c_i(x_k))\nabla^2_{xx} c_i(x_k)]d + \nabla_x \mathcal{L}_A(x_k, \lambda^k; \mu_k)^T d$ subject to $l \leq x_k + d \leq u$ and $||d||_\infty \leq \Delta$ where Δ is the trust-region size and A here is the matrix whose rows are gradients of constraints. The term $\nabla^2_{xx} f(x_k) + \mu_k A^T A - \sum_i(\lambda_i^k - \mu_k c_i(x_k))\nabla^2_{xx} c_i(x_k)$ represents $\nabla^2_{xx}\mathcal{L}_A(x_k, \lambda^k; \mu_k)$. We can solve this approximately using nonlinear gradient projection technique which involves two steps. First, we search for a local optimum called the Cauchy point along the steepest descent direction and bent it every time we hit any of the box boundaries. Then, we fix the active set at the Cauchy point and find a better solution than the Cauchy point using conjugate gradient method for subspace minimization. In this case, we move from the Cauchy point in the direction where we do not change the current active set.

Theoretically, it can be shown that, under some mild assumptions, the local solution of the original problem is a strict local minimizer of $\mathcal{L}_A(x, \lambda^*; \mu)$ where λ^* is the Lagrange multipliers that make the local solution satisfy the second-order sufficient condition and $\mu \geq \bar{\mu}$ for some finite threshold value $\bar{\mu}$.

3 Results and discussion

3.1 AL implementation

In this project, we solve the optimization problem using the technique of augmented Lagrangian. The overall optimization process is similar to Algorithm 17.4 in Nocedal and Wright (2006) in which we use the same update scheme of optimization hyperparameters as that of LANCELOT software package (Conn, Gould, & Toint, 1992). We find the approximate solution of bound-constrained nonlinear subproblem using the gradient projection method with trust-regions as outlined in section 2.2. For the gradient projection process, we follow Algorithm 16.5 in Nocedal and Wright (2006). For the subspace minimization process, we follow the projected conjugate gradient (CG) technique in Algorithm 16.2 of Nocedal and Wright (2006). For simplicity, we use the identity matrix as the preconditioning matrix in the CG iterations. We stop the optimization when the infeasibility measure $||c(x_k)||$ falls below 10^{-4} and the optimality condition $||x_k - P(x_k - \nabla_x \mathcal{L}_A(x_k, \lambda^k; \mu_k), l, u)||$ is smaller than 10^{-3} of the original value.

To frame our optimization problem to suit this AL implementation, we introduce slack variables $b_i^{(j)}$ and $s^{(j)}$ for $i \in \{1, \ldots, N\}$ and $j \in \{1, \ldots T\}$ which corresponds to constraints (3) and (4), respectively. Hence, we can rewrite our optimization problem as

$$max_{t_i^{(j)}, q_{g,i}^{(j)}} \sum_{j=1}^{T} \sum_{i=1}^{N} CGR_i \times q_{g,i}^{(j)} \times \Delta t \times Price^{(j)} \tag{5}$$

$$s.t \quad q_{g,i}^{(j)} \times \Delta t = \frac{q_{pot,i}}{d_i} ln(\frac{1 + d_i t_i^{(j)}}{1 + d_i t_i^{(j-1)}}) \quad \forall j \in \{1, \ldots, T\}, \forall i \in \{1, \ldots, N\} \tag{6}$$

$$t_i^{(j)} - t_i^{(j-1)} - b_i^{(j)} = 0 \quad \forall j \in \{1, \ldots, T\}, \forall i \in \{1, \ldots, N\} \tag{7}$$

$$Q_{nom} - \sum_{i=1}^{N} q_{g,i}^{(j)} \times \Delta t - s^{(j)} = 0 \quad \forall j \in \{1, \ldots, T\} \tag{8}$$

$$0 \leq b_i^{(j)} \leq \Delta t \quad \forall j \in \{1, \ldots, T\}, \forall i \in \{1, \ldots, N\} \tag{9}$$

$$0 \leq s^{(j)} \quad \forall j \in \{1, \ldots, T\}. \tag{10}$$

We have $3NT + T$ variables with $2NT + T$ constraints in total (excluding box constraints). Turning this setup into an augmented Lagrangian function with bound constraints is then straightforward. We only consider the problem with $N = 3$ in this project but the method and code can be used for higher number of wells without any modifications. As indicated earlier, we consider the case where $\Delta t = 1$, $Q_{nom} = 30$ and price model with depreciation factor of 0.9. We perform gradient and Hessian checking using finite differences method to ensure integrity of elements of optimization process as reported in section 5.1 in the appendix. We perform this optimization using MATLAB with only basic linear algebra package.

3.2 Simulation cases and results

We show that the optimization method works perfectly by reporting results of 4 different cases. In each case, we report the well properties including the maximum potential (q_{pot}), decline rate (d), and condensate gas ratio (CGR). We use two types of visualization to report the optimal solution. The first one is the production allocated to each well over different periods while the second one reports the open duration of each well at each timestep.

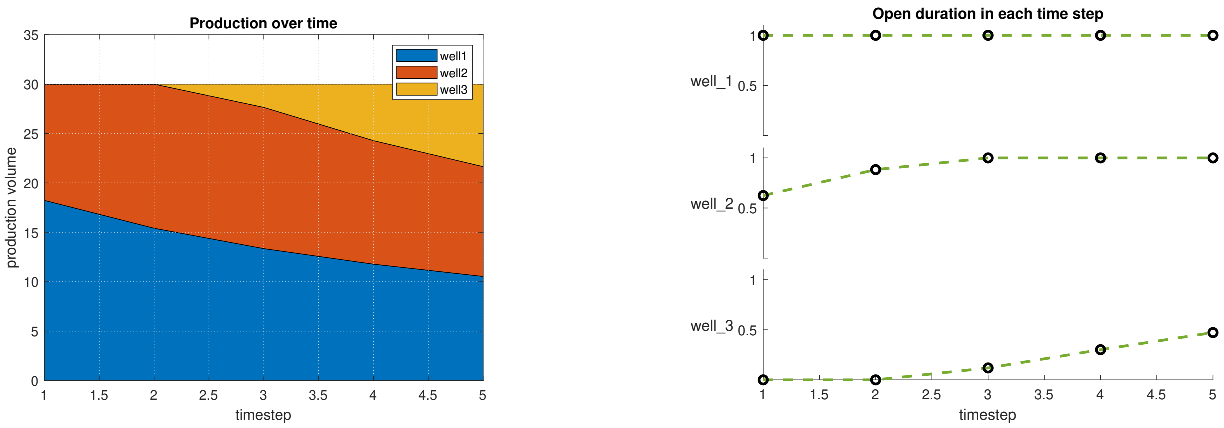
3.2.1 Case I

In case I, we would like to test that the optimal operating plan correctly prioritizes wells with higher CGR values. We report the well properties in Table 1. It is clear that among these three wells, the first one is the one we should prioritize in our production plan as it has the highest CGR while all other parameters are the same as others. If we have a production quota left, then we can consider producing from the second well and the third, respectively.

Table 1: Well properties of case I (T = 5 periods)

	Potential (q_{pot})	Decline rate (d)	CGR
well1	20	0.20	20
well2	20	0.20	11
well3	20	0.20	2

The optimal solution, shown in Figure 1, from our optimization process reflects this correctly. In the first timestep, we open only the first two wells to reach the nomination rate of 30 unit. In fact, we do not even need to open the second well all the time in this first timestep as shown in the plot on the right. After that, when the first two wells keep declining, we open the last one just to make sure we reach our quota.



(a) Production allocation for each well

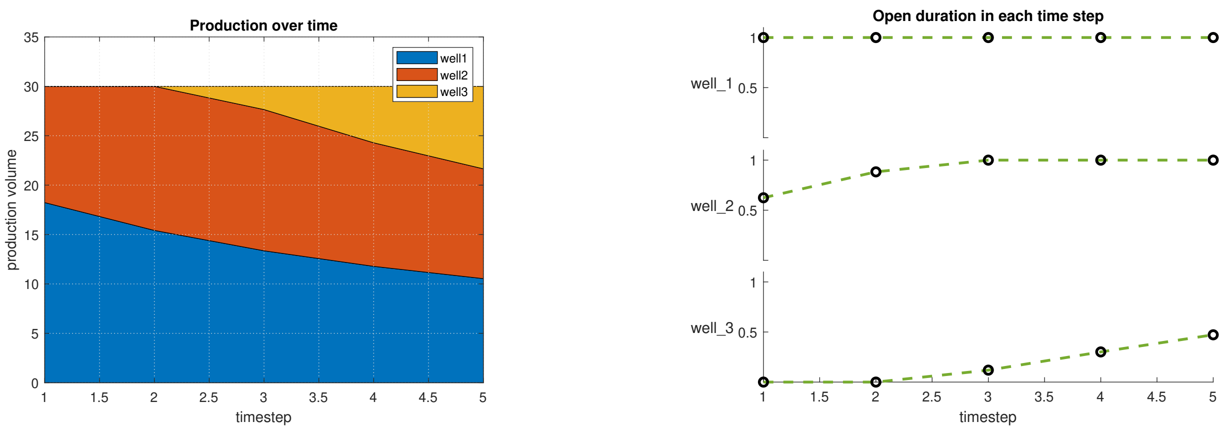
(b) Open duration for each well

Figure 1: Optimization result for case I

3.2.2 Case II

One of the appealing benefits of the AL method is the ability to handle infeasible start naturally. To find a feasible start for some other algorithms, sometimes, practitioners perform a single-period optimization iteratively. This process can have the same order of complexity as the one we are trying to solve and, hence, benefits from using the AL method is immense.

In this case, we use the same well properties as those of case I, but we initialize our starting point x_0 with random vector of appropriate size. As expect, we obtain the same optimal solution, shown in Figure 2, as per case I.



(a) Production allocation for each well

(b) Open duration for each well

Figure 2: Optimization result for case II

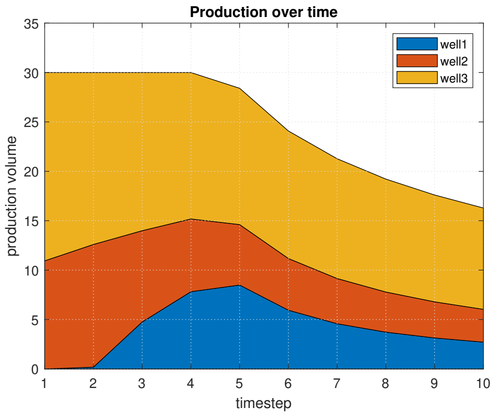
3.2.3 Case III

We study the plateau effect on our optimization. Gas field production has a fixed concession time. Hence, sometimes, we need to prioritize a well that declines slowly because otherwise there would be a lot of unproduced fluids underground at the end of production agreement.

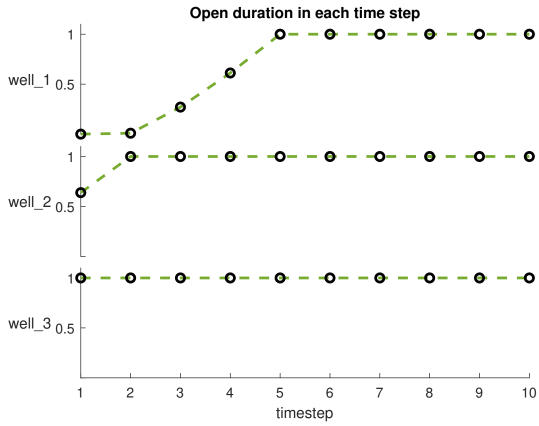
Table 2 reports the well properties in this case. We fix CGR and initial potential of all wells. We make the first well decline at the fastest rate while the third well has the slowest decline rate. Intuitively, as the first well can be depleted in a short time, we should capitalize as much as possible from the third well as, otherwise, there would be an opportunity loss. The result shown in Figure 3 perfectly matches this expected behavior. We open the last well all the time during our production period while we are fulfilling the nomination quota using the second well followed by the short-lived first well.

Table 2: Well properties of case III (T = 10 periods)

	Potential (q_{pot})	Decline rate (d)	CGR
well1	20	1.00	20
well2	20	0.55	20
well3	20	0.10	20



(a) Production allocation for each well



(b) Open duration for each well

Figure 3: Optimization result for case III

3.2.4 Case IV

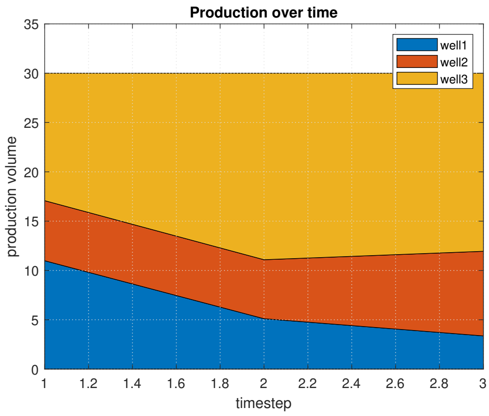
Lastly, in case IV, we consider the trade-off between time value of money and how the concession period can affect the production plan. We describe the well properties in Table 3. All wells have the same potential but with different decline rates and CGR. While the first well has the highest CGR, it declines the fastest among the three. On the contrary, the last well depletes at a slow rate but, at the same time, it has the lowest condensate content.

Table 3: Well properties of case IV

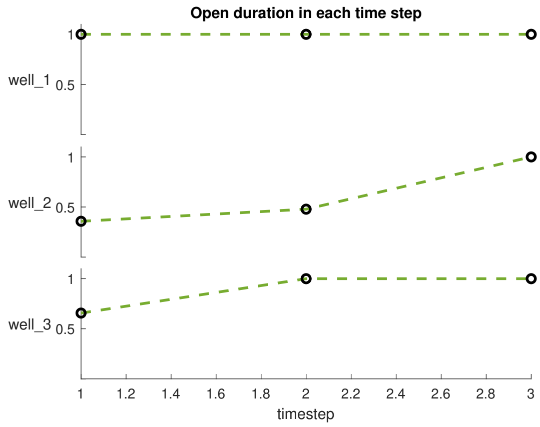
	Potential (q_{pot})	Decline rate (d)	CGR
well1	20	2.00	20
well2	20	1.03	17.5
well3	20	0.05	15

When the field has little time left until the concession ends, it is intuitive to focus on the wells with high CGR. To show this, we run the optimization with $T = 3$ and show the result in Figure 4. Clearly, we open the well with the highest CGR (the first well) all the time.

The reason why we cannot fully prioritize the second well rather than the last one can be explained in term of the production plateau. Even though the second well has higher CGR than the last one, it depletes much faster. Consequently, forcing this second well to open all the time will lower our production in the later periods. We provide the production profile for the case where we force the second well to open along the first all the time in Figure 5. In this case, we cannot reach our nomination target in the last period as the second well dies down already. Hence, we lose condensate selling associated to that gap.

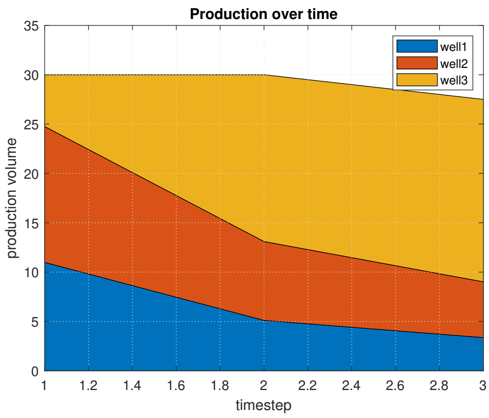


(a) Production allocation for each well

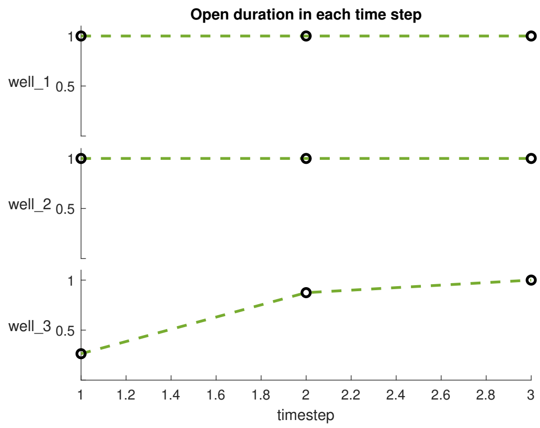


(b) Open duration for each well

Figure 4: Optimization result for case IV (short field life)



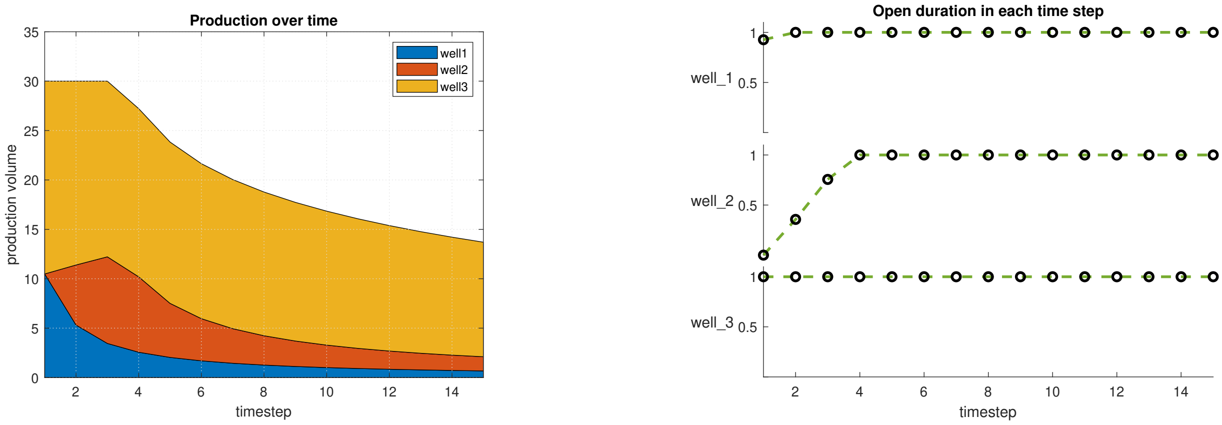
(a) Production allocation for each well



(b) Open duration for each well

Figure 5: Production profile for case IV if well 1 and well 2 are forced to open all the time

For a field with longer concession life, we should open the third well as much as possible as failing to do so would lead to opportunity loss from unproduced condensate underground. The result in Figure 6 in which $T = 15$ shows this production plan. It forces the third well to open all the time. Whenever we cannot reach the production target, we open the first well followed by the second well to optimize the condensate production. Notice that the higher CGR does not make the first well more preferable than the last one even in the first timestep when the value of money is highest. This is where the multi-period optimization shines. In the single-period optimization, we would greedily open the first well for full duration rather than the last well as this gives the most money at the current time. However, this operating plan falls short as this could lead to much lower production in later periods.



(a) Production allocation for each well

(b) Open duration for each well

Figure 6: Optimization result for case IV (long field life)

3.3 KKT condition and convergence analysis

Using the stopping criterion mentioned above, for all of our scenarios, our solutions satisfy the first-order optimality condition. The values of $||x^k - P(x^k - \nabla_x \mathcal{L}_A(x^k, \lambda^k; \mu_k), l, u)||$ are on the order of 10^{-2} to 10^{-3}. As this measure depends on the size of x, by normalizing it with the dimension of x, the KKT condition is satisfied to the order of 10^{-3} to 10^{-4}. Of course, forcing this KKT error down further is possible with longer run time but we observe that we can get the answer with high level of accuracy already with the stated criterion.

It is not straightforward to evaluate the convergence rate for problems with inequality constraints. However, we can check the convergence of the trust-region method that we use to solve the AL subproblem. As this is not the main focus, this result is reported in section 5.2 of the appendix. We obtain a convergence that looks better than a linear convergence.

4 Concluding remarks

We successfully show that we can solve the multi-period gas field optimization problem with Harmonic decline with the augmented Lagrangian technique. We show that our method works perfectly under different scenarios. This work suggests possibility of replacing the traditional single-period linear optimization with linearized constraints with this nonlinear optimization method. We note that we can generalize this paradigm to wells with other types of decline patterns easily by substituting the gradients and Hessians of Harmonic decline constraints with those of other patterns. Moreover, adding other constraints typically adopted in the production process such as the contamination constraints (CO_2 or Hg) is trivial as these are linear behavior.

Future work could focus on testing large-scale optimization result where the number of wells and periods is large. Also, as several gradients and Hessian matrices are sparse, the sparse linear algebra operations are worth considering.

References

Arps, J. (1945, 12). Analysis of Decline Curves. *Transactions of the AIME*, *160*(01), 228-247. Retrieved from https://doi.org/10.2118/945228-G doi: 10.2118/945228-G

Conn, A. R., Gould, N. I. M., & Toint, P. L. (1992). *Lancelot: A fortran package for large-scale nonlinear optimization (release a)*. Berlin, Heidelberg: Springer-Verlag.

Lo, K., Starley, G., & Holden, C. (1995, 02). Application of Linear Programming to Reservoir Development Evaluations. *SPE Reservoir Engineering*, *10*(01), 52-58. Retrieved from https://doi.org/10.2118/26637-PA doi: 10.2118/26637-PA

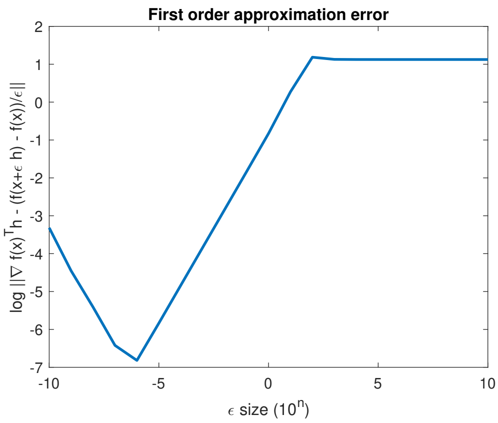
Nocedal, J., & Wright, S. (2006). *Numerical optimization* (2. ed. ed.). New York, NY: Springer. Retrieved from http://gso.gbv.de/DB=2.1/CMD?ACT=SRCHA&SRT=YOP&IKT=1016&TRM=ppn+502988711&sourceid=fbw_bibsonomy

Wongkamthong, C., Wongpattananukul, K., Suranetinai, C., Vongsinudom, V., & Ekkawong, P. (2018, 10). *In-House Software Development for Gas Production Optimization: A South East Asia Perspective* (Vols. Day 2 Wed, October 24, 2018). Retrieved from https://doi.org/10.2118/192080-MS (D021S012R003) doi: 10.2118/192080-MS

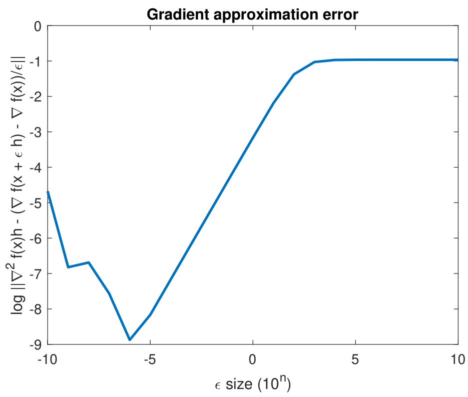
5 Appendix

5.1 Gradient checking

We check gradient and Hessian calculations of each part of the augmented Lagrangian function using finite differences method. Here, for the Harmonic decline constraints, we report the error from using first order approximation of the function value $||\nabla f(x)^T h - (f(x+\epsilon h) - f(x))/\epsilon||$ with different sizes of ϵ in the left subfigure of Figure 7. We also report the error from approximating gradients with Hessian $||\nabla^2 f(x)h - (\nabla f(x+\epsilon h) - \nabla f(x))/\epsilon||$ in the right subfigure. They both exhibit the behavior we should expect from this highly nonlinear function.



(a) Gradient checking by using it to approximate function values



(b) Hessian checking by using it to approximate gradients

Figure 7: Checking gradient and Hessian calculations of the Harmonic decline constraint

5.2 Convergence of trust-region to solve subproblem

To show that the gradient projection with trust-region method used to solve the AL subproblem works perfectly, we report, in one of our iteration of AL, the KKT condition measure $||x^k - P(x^k - \nabla_x \mathcal{L}_A(x^k, \lambda^k; \mu_k), l, u)||$ over each step in the trust-region iteration in Figure 8. As we mentioned in the main text, the convergence takes the form that seems to be better than the linear convergence as expected.

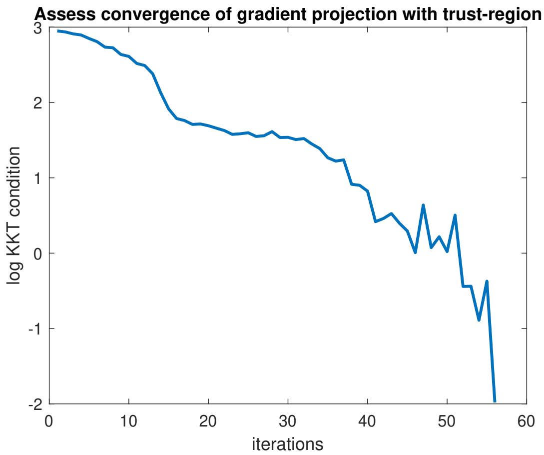


Figure 8: Convergence of trust-region method to solve AL subproblems

5.3 File descriptions

- `AL_main.m` : Main MATLAB script to run simulations for different scenarios.
- `computeKKT_AL.m` : Function to evaluate the optimality condition.
- `gen_case_1.m` : Function to generate an instance of case I problem.
- `gen_case_2.m` : Function to generate an instance of case II problem.
- `gen_case_3.m` : Function to generate an instance of case III problem.
- `gen_case_4.m` : Function to generate an instance of case IV problem.
- `gradient_checking.m` : MATLAB script to perform gradient checking for different constraints and objective function using the finite differences technique.
- `gradient_checking_AL.m` : MATLAB script to perform gradient checking for the augmented Lagrangian function using the finite differences technique.
- `plot_q.m` : Function to generate the plot of production contribution over time.
- `plot_t.m` : Function to generate the plot of the open duration over time.
- `project.m` : Function to perform projection onto box constraints $P(x, l, u)$.
- `solution_case_1.mat` : A solution of an instance of case I problem.
- `solution_case_2.mat` : A solution of an instance of case II problem.

- `solution_case_3.mat` : A solution of an instance of case III problem.
- `solution_case_4_long.mat` : A solution of an instance of case IV problem with long field life ($T = 15$).
- `solution_case_4_short.mat` : A solution of an instance of case IV problem with short field life ($T = 3$).
- `gradHess\ALagrangian.m` : Function to evaluate the augmented Lagrangian function together with its gradient and Hessian matrices.
- `gradHess\combineConst.m` : Function to get and combine constraints and their gradients and Hessian matrices.
- `gradHess\computeHarmonicConstr.m` : Function to evaluate the Harmonic decline constraints together with its gradient and Hessian matrices.
- `gradHess\computeNomConstr.m` : Function to evaluate the nomination constraints together with its gradient and Hessian matrices.
- `gradHess\computeObjGradHess.m` : Function to evaluate the objective function together with its gradient and Hessian matrices.
- `gradHess\computeTimeConstr.m` : Function to evaluate the time constraints together with its gradient and Hessian matrices.
- `TR_ALsubproblem\calculate_t_bound.m` : Function to calculate t until we can move to hit each of the boundaries in the negative gradient direction in the gradient projection technique in order to find the Cauchy point.
- `TR_ALsubproblem\CG_subproblem.m` : Function to perform subspace minimization to find a better solution than the Cauchy point. This is used in the second step of the gradient projection technique to solve the AL subproblem.
- `TR_ALsubproblem\getActiveSet.m` : Function to get the active set indices from the current iterate/Cauchy point.
- `TR_ALsubproblem\getCauchypoint.m` : Function to find the Cauchy point in the gradient projection technique.
- `TR_ALsubproblem\solveWithTR.m` : Function to solve the AL subproblem using the gradient projection with trust-region.