Modified Dynamic Programming Algorithm and Its Application in Distribution of Power Plant Load

Wanshan Wang¹, Peihong Wang¹* and Yihua Dong²,³

¹ School of Energy and Environment, Southeast University, Nanjing, Jiangsu Province, 210096, China
² Zhejiang Energy Technology Research Institute Co. Ltd, Hangzhou, Zhejiang Province, 311121, China
³ Key Laboratory of High Efficiency Energy Conservation and Pollutant Control Technology of Thermal Power Generation, Zhejiang Province, Hangzhou, Zhejiang Province, 311121, China

Abstract. Dynamic programming algorithm is the classic method in distribution of plant loads. The traditional calculation method of dynamic programming is easy to make the calculation take too long time. This paper proposes two modified methods of dynamic programming, namely, dynamic programming in progressive structure and dynamic programming in progressive step-size. What is more, these two methods are combined to a new hybrid method. And these three methods and conventional dynamic programming are applied to the distribution of thermal power plant load to verify. The results of modified methods contrast with the conventional method show the modified methods reduce the calculation time and alleviate the curse of dimensionality effectively, ensuring the accuracy of calculation.

1 Introduction

Dynamic programming is a branch of operations research and a mathematical method to achieve the optimization of multi-decision processes. With its good performance and applicability, it has been widely used in many fields such as power systems [1-3], but if the number of units involved in the plant is large, the calculation complexity is very large, taking quite long time to finish the calculation. The shortcoming makes it difficult to achieve fast distribution of the load. Fukao et al. [4] applied dynamic programming to power system optimization operations. Najafi [5] proposed a new harmonic search algorithm for unit combination problem, which greatly improved the computational efficiency; Ren et al. [6] proposed a dynamic programming method based on the priority order method, which is suitable for the systems of small units. However, for large-scale systems with a large number of units, the calculation amount is very large, and the optimal solution may not be found. The research optimized the dynamic programming. Although the computational efficiency of the method is improved, and the complexity of the algorithm is increased, and the versatility and reliability are reduced.

In order to improve the calculation speed, this paper sets up the general model of dynamic programming for distribution of power plant load, and analyzes the calculation process of dynamic programming, and introduces the structural progressive method (SPM) and step-size progressive method (SSM) and combines them into a hybrid progressive method (HPM) to modify the dynamic programming. Taking a thermal power plant as the research object, the results of modified methods and conventional method (CM) were compared. It shows that the modified methods improve the calculation speed while ensuring the accuracy.

2 Analysis of Dynamic Programming for Distribution of Power Plant Load

2.1 General model of Dynamic Programming for Distribution of Power Plant Load

Stage: n stages for a plant with n units;
State: set P as the state variable of stage I to represent the total load of the former i numbers of units;
Decision variable: set Xi as the decision variable of stage I, which represents the load of the i-th unit
State transition equation:

\[ P_{i+1} = P_i + X_{i+1} \] (1)

Boundary condition:

\[ P_0 = 0 \] (2)

Optimal value function: the minimized standard coal consumption when the total load of the former i unit is P:

\[ F_i(P_i) = \min_{X_{i,\ldots,X_{i-1}}} \sum_{j=1}^{i} F_j(X_j) \] (3)

Decision set:

\[ G_i(X_i) = \{ X_i \mid X_{i,\min} \leq X_i \leq X_{i,\max}, P_{i+1} + X_i = P_i \} \] (4)

Where \( X_{i,\min} \) and \( X_{i,\max} \) are the minimum value and maximum value of the \( X_i \) respectively.

Recursive equation:

*Corresponding author’s e-mail: phwang@seu.edu.cn
\[ F_i(P_i) = \min_{X_i} \left\{ f_i(X_i) + F_{i-1}(P_i) \right\} \]  
(5)

2.2 Model of Dynamic Programming for Distribution of Power Plant Load

2.2.1. Sequential table.

First step:
\[ \begin{align*}
F_1(P_1) &= f_1(X_1) \\
P_1 &= X_1 \\
X_{1_{\min}} &\leq X_1 \leq X_{1_{\max}}
\end{align*} \]
(6)

\( P_I \) traverses its value interval \([X_{1_{\min}}, X_{1_{\max}}]\) in a certain step-size.

Intermediate step:
\[ \begin{align*}
F_j(P_j) &= \min_{P_j} \left\{ f_j(X_j) + F_{j-1}(P_{j-1}) \right\} \\
P_j &= P_{j-1} + X_j \\
X_{1_{\min}} &\leq X_j \leq X_{1_{\max}}
\end{align*} \]
(7)

\( P_j \) is traversed in a certain step-size, and the calculation results of the previous stages are combined to calculate the optimal value of the coal consumption \( X_j \) corresponding to \( P_j \), and the data is recorded.

2.2.2. Reverse order.

Corresponding to the total load \( P_{\text{total}} \) of the whole plant, in the \( n \) stage, the optimized load \( X_n \) of the \( n \)th unit can be directly found from the table: For the \( n \)-th stage, the corresponding optimization load is queried by \( P_{n-1} = P_{\text{total}} - X_n \). Like this, the distribution of load can be determined according to the state transition equation.

3 Modifying of Dynamic Programming

With the fine of the step-size and the increase of the number of units, the computational complexity of dynamic programming has grown exponentially, and the problem, the curse of dimensionality, is extremely prominent [7]. The dynamic programming method divides a given original problem into several sub-stages or several sub-problems, and then solves them in turn. The optimal solution combination of all sub-stages or sub-problems is the optimal solution of the original problem. That is, an overall optimization problem can be decomposed into a sequence multi-stage optimization problem to solve.

In order to relieve the curse of dimensionality, two modified methods of dynamic programming are proposed, namely, structural progressive dynamic programming and step-size progressive dynamic programming.

3.1 Dynamic programming in progressive structure

Dynamic programming is to divide an original process into multiple sub-stages (denoted as stage 1, stage 2, stage 3, stage 4, stage 5, ..., stage \( n \)), and consider the stage1 and stage 2, and get the best combination scheme set (noted as A) for stage 1 and stage 2. The scheme set A can be equivalent to a new stage, combined with stage 3 to obtain a new equivalent stage B, and so on. The final step is the combination of unit \( n \) and the equivalent stage of the previous \( n-1 \) stages to obtain the most optimal distribution of the whole process finally.

The structural progressive method is to divide an original process into multiple sub-stages (denoted as stage 1, stage 2, stage 3, stage 4, stage 5, ..., stage \( n \)), and select several stages to combine, obtaining the equivalent stage A1. In the remaining stages, several stages are selected to combine, getting the equivalent stage A2, and so on until all the sub-stages are combined. Several first-level equivalent stages (A1, A2, ..., Ai) are obtained. Then the similar process of the first-level equivalent stages is carried out to obtain the several second-level equivalent stages (B1, B2, ..., Bj), and most optimal distribution of the whole process is obtained by analogy. This paper use the combination of two to refer to the structural progressive method.

The schematic diagram of the dynamic programming applied in load distribution is shown in Figure 1. The schematic diagram of the structural progressive dynamic programming applied in load distribution is shown in Figure 2.
3.1.1. Sequential table. There are \( n \) units in total. If \( n \) is even, there would be \( n/2 \) first-level equivalent stages. If \( n \) is odd, there are \( n/2+1 \) first-level equivalent stages.

First step - combination of sub-stages.

For \([n/2]\) \( [x] \) means the rounding up for \( x \), the below is same), combination of sub-stages is:

\[
F_i(P_i) = \min_{P_j} \left\{ F_{i,j}(P_{i,j}) + F_{i,j+1}(P_{i,j+1}) \right\}
\]

\[
P_{i,j} = P_{i,j+1} + P_{i,j+2}, \quad 1 \leq i \leq n/2 \tag{8}
\]

\[
F_{i,j}(P_{i,j}) = f_{i,j}(P_{i,j}), \quad F_{i,j+1}(P_{i,j+1}) = f_{i,j+1}(P_{i,j+1})
\]

\[
X_{i,1 \text{min}} \leq P_{i,1} \leq X_{i,1 \text{max}}, X_{i,2 \text{min}} \leq P_{i,2} \leq X_{i,2 \text{max}}
\]

Where \( F_i(P_i) \) is the relation of input and output in the \( i \)-th equivalent stage of first-level, stage \( 1 \), \( F_{1,1}(P_{1,1}) \) and \( F_{1,2}(P_{1,2}) \) are the the relation of input and output in the two selected stages to combine, stage \( 1,1 \) and stage \( 1,2 \), which are the sub-stages.

\( P_{i,1} \) traverses its value interval \([X_{i,1 \text{min}}, X_{i,1 \text{max}}]\) in a certain step-size; \( P_{i,2} \) traverses its value interval \([X_{i,2 \text{min}}, X_{i,2 \text{max}}]\) in a certain step-size.

If there is \([n/2]+1\) equivalent stage, then there is

\[
F_i(P_i) = f_{i,1}(P_i)
\]

\[
X_{i,\text{min}} \leq P_i \leq X_{i,\text{max}}, \quad i = 1 + n/2 \tag{9}
\]

\( P_{i,1} \) traverses its value interval \([X_{i,1 \text{min}}, X_{i,1 \text{max}}]\) in a certain step-size.

Intermediate step, for a certain middle level, \( j \) = [the number of the \((m-1)\) level equivalent stages /2], the combination is:

\[
F_m(P_m) = \min_{P_{m,i}} \left\{ F_{m,i}(P_{m,i}) + F_{m,i+1}(P_{m,i+1}) \right\}
\]

\[
P_{m,i} = P_{m,i+1} + P_{m,i+2}, \quad 1 \leq i \leq j \tag{10}
\]

Where \( F_m(P_m) \) is the relation of input and output in the \( m \)-level equivalent stage at \( m \)-level, stage \( m,1 \) and \( F_{m,2}(P_{m,2}) \) are the the relation of input and output in the two selected stages to combine, stage \( m,1 \) and stage \( m,2 \), which are the last level equivalent stages.

If number of the equivalent stages in last level is odd, the unpair stage is setted as the next level equivalent stage directly.

The final step, it can be denoted by \( z \):

\[
F_z(P_z) = \min_{P_{z,i}} \left\{ F_{z,i}(P_{z,i}) + F_{z,i+1}(P_{z,i+1}) \right\}
\]

\[
P_z = P_{z,1} + P_{z,2}
\]

Where \( F_z(P_z) \) is the relation of input and output in the whole process. \( F_{z,1}(P_{z,1}) \) and \( F_{z,2}(P_{z,2}) \) are the the relation of input and output in the two selected stages to combine, stage \( z,1 \) and stage \( z,2 \), which are the last level equivalent stages.

3.2 Dynamic programming in progressive step-size

The step-size progressive method of dynamic programming draws on the idea of differential dynamic programming. Differential dynamic programming is a modified method of dynamic programming, which is proposed to solve the curse of dimensionality in conventional dynamic programming. The principle is to determine a decision that is as close to the most optimal decision as possible based on experience or other simple methods and obtain a corresponding initial state sequence; a small range (increment) is changed on the initial state sequence to form a corridor. In the corridor, the conventional dynamic programming solution is used to optimize. The step-size progressive method of dynamic programming is to determine the possible interval where the optimal strategy may exist by selecting a large step-size, and then select a smaller step-size to determine a new possible interval within the possible interval selected before, sequentially reciprocating, and finally get the optimal strategy.
3.2.1. Sequential table. The first step: select a specific step-size of dynamic programming to optimize the whole plant and obtain the load distribution under the specified total load. According to the obtained load distribution, select the possible interval of the optimal state of each stage, and record it.

Intermediate step: select a smaller step size than before and use the dynamic programming method to find the updated load distribution from the possible interval selected in the last step. According to the obtained new load distribution, the possible interval is updated.

Final step: select a specific smallest step size, and use the dynamic programming method to obtain the updated load distribution under the specified total load from the possible interval selected in the last step. This is the final load distribution.

3.3 Dynamic programming in progressive hybrid

In step-size progressive method of dynamic programming, if the possible interval selected is too large, the calculation time will be very long; if the possible interval selection is too small, it is possible that the optimization accuracy is not enough. Therefore, dynamic programming in progressive hybrid is adopted.

The first step: select a specific step size, and use the dynamic programming method to optimize the whole plant, and obtain the load distribution under the specified total load.

The Second step: combine each two stage as a new equivalent stage. If the number of units is odd, the unpaired would be a new equivalent stage. According to the load distribution scheme obtained in the first step, the appropriate range is selected as possible interval of each state. Based on the feasible range of the subunit's load, dynamic programming is performed to determine the load distribution schemes in the possible interval of equivalent stage.

Intermediate step: combine the equivalent stages of the last step, determine the possible interval of the new equivalent stage based on the possible interval of the stage which is combined, and get the load distribution in the possible interval of new equivalent stage.

Final step: based on the possible interval of the sub-combination, the load distribution under the specified total load is obtained. This is the final load distribution.

4 The calculation and analysis of an engineering example

There are 10 units in a power plant, and the quadratic polynomial \( f_i(P) = aP^2 + bP + c \) is selected as the fitting function of the standard coal consumption (CC) and power supply (P). The characteristic data of 10 units obtained in a certain period of time is shown in Table 1.

| \( P_{\text{max}} \) | \( P_{\text{min}} \) | \( a \) | \( b \) | \( c \) |
|----------------|----------------|------|------|------|
| 60 | 15 | 0.00510 | 2.2034 | 15.0 |
| 80 | 20 | 0.00396 | 1.9101 | 25.0 |
| 100 | 30 | 0.00393 | 1.8518 | 40.0 |
| 120 | 25 | 0.00382 | 1.6966 | 32.0 |
| 150 | 50 | 0.00212 | 1.8015 | 29.0 |
| 280 | 75 | 0.00261 | 1.5354 | 72.0 |
| 320 | 120 | 0.00289 | 1.2643 | 49.0 |
| 445 | 125 | 0.00480 | 1.213 | 82.0 |
| 520 | 250 | 0.001276 | 1.1954 | 105.0 |
| 550 | 250 | 0.00135 | 1.1284 | 100.0 |

At the same computing environment, the calculations of the instance are as follows.

1MW step-size for the above ten units is used to sequential table in the full load range. Using conventional dynamic planning takes 13.987s, while the structural progressive method of dynamic planning takes 2.515s. And the calculation results of two methods are exactly the same. The results show that structural progressive method achieves the same accuracy with one-fifth of the conventional calculation time. For the specific load 2000MW, the informations of calculations are shown in Table 2.

| Step-size | CM | SPM | SSM1 | SSM2 | HPM |
|-----------|----|-----|------|------|-----|
| First step-size | 1 MW | 1 MW | 5 MW | 5 MW | 5 MW |
| Second step-size | 1 MW | 1 MW | 10 MW | 50 MW | 50 MW |
| Possible interval | 10.398s | 1.567s | 5.111s | 8.299s | 1.226s |
| Costs time | 4688.4 | 4688.4 | 4688.8 | 4688.6 | 4688.5 |
5 Conclusions

This paper proposed two modified methods of dynamic programming, dynamic programming in progressive structure and dynamic programming in progressive step-size. Those two methods are combined into dynamic programming in progressive hybrid. From the applications of the methods in distribution of power plant load, the results show dynamic programming in progressive structure reduces the calculation time while ensuring the same calculation results. The calculation amount is greatly reduced, and the calculation speed is obviously improved. Dynamic programming in progressive step-size could also reduce the amount of calculation and increase the calculation speed, but there are problems that the calculation accuracy is not enough and the most optimization may not be obtained. The dynamic programming in progressive hybrid is used to improve the calculation speed further, and the calculation accuracy is compensated to a certain extent.

Although the versatility of dynamic programming in progressive structure has yet to be proved by further work, its combination with dynamic programming in progressive step-size is sufficient to prove that the method could be used in more complex scenarios.

References

1. Yeh, W. W. G. (1985). Reservoir management and operations models: A state of the art review. Water resources research, 21: 1797-1818.
2. Yi, J., Labadie, J. W., Stitt, S. (2003). Dynamic optimal unit commitment and loading in hydropower systems. Journal of Water Resources Planning and Management, 129: 388-398.
3. Bellman, R. (1966). Dynamic programming. Science, 153: 34-37.
4. Fukao, T., Yamazak, T., Kimura, S. (1959). An application of dynamic programming to economic operation problem of a power system. Electric Technical Journal of Japan, 5, 64-68.
5. Najafi, S. (2012). A new heuristic algorithm for unit commitment problem. Energy Procedia, 14: 2005-2011.
6. Ren, Y., Zhang, X. Q. (2010). Research of unit commitment in power system based on improved dynamic programming method. Tech Autom Appl, 29: 6-8.
7. Feng, Z. K., Niu, W. J., Cheng, C. T., Liao, S. L. (2017). Hydropower system operation optimization by discrete differential dynamic programming based on orthogonal experiment design. Energy, 126: 720-732