The Sydney Journal of Interdisciplinary Engineering
SJIE 1(3) X–X (2025) https://doi.org/XXXX/XXXX

From the Yangtze to the Gobi Desert: Designing and Evaluating a Proposed Pipeline for Combatting Desertification

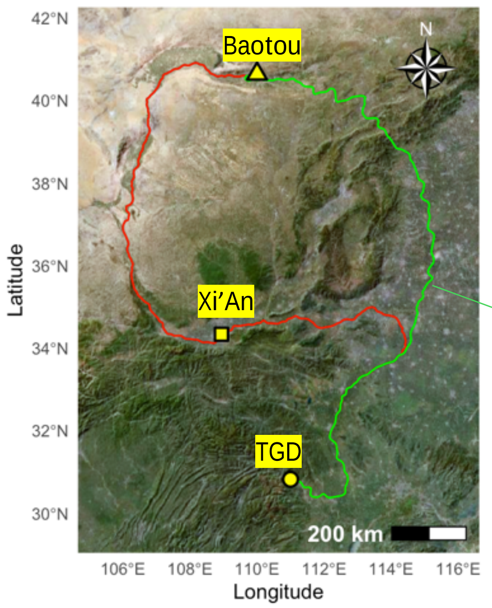
Osama Rehman[1], Jonathan Tapia[1], Matthew Wolfenden[1], and Karinna Yee[1], Gobinath Rajarathnam[1]

[1] School of Chemical and Biomolecular Engineering, Building J01,
The University of Sydney, Camperdown, NSW 2006, Australia

E-mail: oreh8989@uni.sydney.edu.au, jtap7186@uni.sydney.edu.au, mwol4185@uni.sydney.edu.au, kyee2383@uni.sydney.edu.au

Received xxxxxx
Accepted for publication xxxxxx
Published xxxxxx

Graphical Abstract

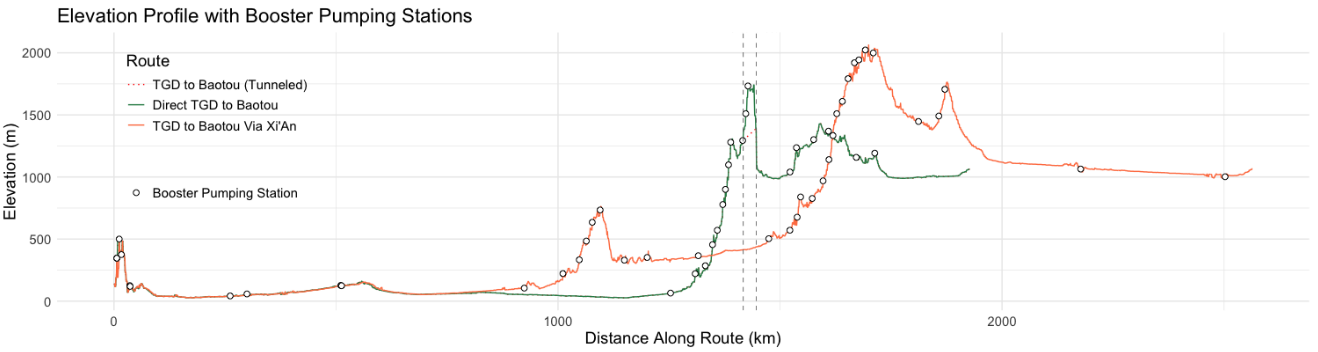


Three pipelines were assessed:

A. Direct route from Three Gorges Dam to Baotou with tunnel
B. Direct route from Three Gorges Dam to Baotou
C. Indirect route via Xi'An

Optimal pipeline (B. Three Gorges Dam to Baotou):

- Nominal diameter: 2000 mm
- Flow: 2 m^3/s
- Length: 1927 km
- Social impact: 400,000,000 lives improved annually
- Environmental impact: 3,600 km^2 p.a. of desertification reversal
- Construction cost: US$ 8 billion
- Lifetime operational cost: US$ 12 billion



SJIE 1(3) X–X (2025) https://doi.org/XXXX/XXXX

Abstract

This study proposes and evaluates a transregional water pipeline from the Three Gorges Reservoir (TGR) to Baotou at the Gobi Desert's southern edge, aiming to combat desertification in northern China which affects up to 400,000,000 people and costs an estimated US$ 188 billion to China's GDP. Optimal pipeline pathfinding and hydraulic design was conducted for a 2 m diameter pipeline carrying 173 ML/day of water, and booster pumping stations were placed along the route as per head loss requirements. The optimal route was determined to be 1927 km long with 23 booster pumping stations for a power consumption of ~41 MW. Analysis of seasonal precipitation and temperature patterns highlighted concerns about flooding and extreme temperature fluctuations, with central to northern China experiencing temperatures as low as -15 °C in winter and as high as 34 °C in summer. A comprehensive treatment system was designed to ensure resilience against climactic stressors, including large to fine particle filtration, microbial filtration, UV protection, and pH corrosion mitigation strategies. Moreover, by utilizing the hydroelectric and solar energy available along the pipeline route, annual carbon emissions can be reduced by a factor of ten compared to non-renewable energy sources. Tunnelling was considered to minimise energy costs associated with pumping over mountains but was found to be more expensive. The final cost of the Project is estimated to be US$ 20 billion of which US$ 8 billion is the Capital Expenditure and US$ 12 billion is the Operational expenditure.

Keywords: climate resilience, desertification reversal, pipeline design, risk modelling

1. **Introduction/Literature Review**

1.1 Desertification in Northern China

Desertification is land degradation characterised by salinisation, soil erosion, loss of organic matter, depletion of nutrients, and compaction, leading to food insecurity, dust storms, and climate related displacement[1]. It is estimated to cost China 1% of its annual GDP[2] equating to ~US$ 188 billion in 2024[3] and impacts up to 400,000,000 people annually[4,5]. China's ongoing efforts in reverse desertification[6,7] have had a demonstrable improvement[8] including increased vegetation cover, increased biodiversity, and improved soil quality with biocrust formation[9].
Early stages of desertification reversal require significant artificial irrigation (290-340 m^3 water per m^2 of reforested land per year[10]), which, when neglected, results in further groundwater depletion[11].
Degraded land restoration has shown to generate significant socio-economic benefits, ranging from US$ 3 – 6 returns per USD spent over a 30-year period[12].
The Gobi Desert's expansion along the Hexi Corridor has been attributed to groundwater depletion in oasis regions[13] – in some cases directly linked to industrial water users (making up 17.7% of total water consumption in China[14]) such as the Bayan Obo Mine near the city of Baotou which borders the Gobi Desert[15].

1.2 Orographic Water Divide & The Yangtze River

The arid region encompassing the Gobi Desert is bordered by the Dabashan and Qinling Mountain ranges to the south, forming an orographic water divide beyond which are areas of ample precipitation[16].
Below this orographic divide is the Yangtze River – an abundant source of water with an annual mean discharge of approximately 30,000 m^3/s[17].
However, it has a relatively high sediment load and contains significant dissolved nutrients (nitrates & phosphates) and trace elements[17]. The Three Gorges Dam (TGD) impoundment resulted in sediment settling in the Three Gorges Reservoir (TGR) with the trade-off of higher nutrient enrichment[18].
Here, the TGR is considered as a source of water to be transported for desertification reversal.

1.3 Pipeline Megaproject

The Gobi sits on a plateau ~1000 metres higher than TGD, separated also by the Qinling Mountain Range with peaks some 3000 m above sea level. Gravity based water transport (by means of pipeline or aqueduct) are therefore impossible. The difficult terrain also provides significant challenges with road and rail transportation.
This project assumes a high-pressure water pipeline as a design basis. The straight-line distance between Baotou and the TGR is 1100km – among the longest pipelines in operation as of 2025[19]. Furthermore, the Yangtze's water quality regarding nutrient enrichment and pH issues requires specific infrastructure design nuances regarding material selection and water treatment [18].

Moreover, the pipeline design must address various climate challenges along its route from central to northern China. Factors such as, temperature extremities and fluctuations, flooding, and seismic activity must all be accounted for to ensure the operational reliability, structural integrity, and long-term longevity of the pipeline.

2. **Methodology**

2.1 Assumptions

Pipeline design criteria are summarised in Table 1.

Table 1: Summary of Pipeline Design Criteria.

Variable	Assumption	Justification	Reference
Water Source	TGD / TGR	Lowest sediment levels in the Yangtze[18].	Section 1.1
Water Sink	Baotou	Major population centre bordering the Gobi Desert with a large industrial base[15].	Section 1.1
Design Life	50 years	Expected pay-off period of 30 years[20].	Appendix
Flow Rate	2 m^3/s	Calculated based on desertification rate and Baotou water demand.	Appendix

2.2 Pathfinding Methodology

Pipeline routing, plots, and calculations were performed using R via R Studio. An R script for pathfinding was developed to use the NASA ACE2 Digital Elevation Model (DEM)[21].
R code philosophy was as follows:

1. Route start and end latitude / longitude co-ordinates were defined.
2. A bounding box was defined larger than the start and end co-ordinates. The DEM was only processed in this bounding box.
3. A cost matrix was generated for the bounding box using the equivalent ratio (k) between static head (elevation) and frictional pressure drop over pipe length (Pythagorean distance).
4. Path was computed and exported as a path file (.GPX) for processing.
5. Path distance, cumulative elevation gain, total elevation gain, frictional and static pressure drops were computed in a separate script.
6. Tunnels were computed by manually selecting tunnel start and end points and manually modifying the route path. **NOTE**: route files typically overlay a 2D route over a known elevation model. As such, the tunnels are unable to be computed directly in the pathfinding script.

2.3 Booster Pumping Station Placement

Booster pumping stations were placed along the pipeline to ensure the total head loss between stations does not exceed the allowable pump head capacity.
At each segment along the route, the cumulative head loss h_loss is computed as the sum of frictional and elevation-induced losses:

$$h_{loss} = \Sigma [(P_{friction} + P_{elevation}) / (\rho * g)]$$

Where $P_{friction}$is the pressure drop due to friction, $P_{elevation}$ is the pressure required to overcome elevation gain, ρ is the fluid density (1000 kg/m^3), and g is gravitational acceleration (9.81 m/s^2).
A new pumping station is inserted whenever h_loss ≥ H_max, where H_max is the maximum allowable pump head (160 m). After each station placement, the cumulative loss counter is reset.

2.4 Pathfinding Cost Function

To develop a factor (k) for the pathfinding cost function along a pipe on a 45° slope:

$$\Delta P_f = f \cdot \frac{L}{D} \cdot \frac{\rho v^2}{2}$$

$$\Delta P_s = \rho g \Delta z = \rho g L \cdot \sin(45°)$$

Where ΔP_f is frictional pressure drop, $f$is the friction factor, L is the equivalent length of pipe, D is the diameter, and ΔP_s is the static pressure drop.
The ratio then becomes:

$$\frac{\Delta P_f}{\Delta P_s} = \frac{f v^2}{g D^2}$$

With f = 0.01,v = 0.6366 m/s,D = 2 m,g = 9.81:

$$k = \frac{\Delta P_f}{\Delta P_s} \approx 0.00015$$

2.5 Pumping Station Design & Placement

The selected pumps were inline centrifugal water pumps with a maximum head of 160 m and a maximum flow rate of 2400 m^3/h[22] – 3 pumps were required for the pipeline design flow of 2 m^3/s.

Each pump was designed to be independently isolated for automatic switchover and have a check valve on the outlet to prevent backflow in steep sections of pipe.
Variable speed drives (VSD) were selected to allow for slow pump ramp ups on startup and a lower minimum flow than continuous speed drives.
Full redundancy was selected to allow for greater uptime; therefore, each pumping station contains 6 pumps in a 6x33% redundant configuration (Figure 1).
Pressure – vacuum relief valves (P-VRV) were implemented on pump suction to mitigate against high- and low-pressure surges upstream of the pumping station.
Pressure relief valves (PRV) were placed on the pump discharge line to protect the pipeline infrastructure from a pressure excursion.
Bypass flow control was selected to allow for a lower minimum flow delivery[23] (as per fluctuating irrigation demands).

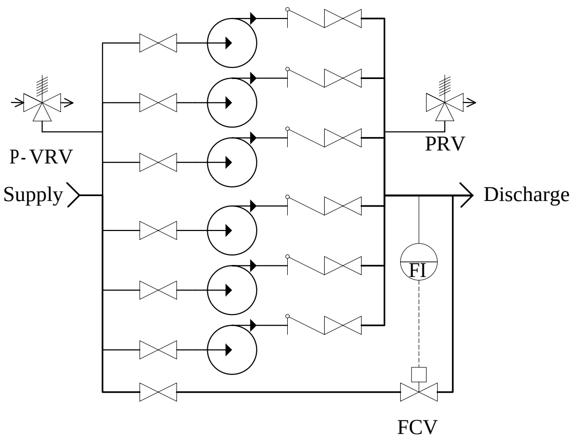


Figure 1: Preliminary booster pumping station design indicating 6x33% redundancy and bypass flow control. Pressure – vacuum relief valves on pump suction, pressure relief on pump discharge.

Booster pumping stations were also determined to include a pipeline isolation valve on the higher-elevation side to allow for maintenance with minimal water losses.
This booster pumping station design is compact and may be housed in prefabricated structures (such as within shipping containers) for rapid on-site installation.
The net booster pumping station cost was estimated to be US$ 1,045,000 as seen in Table 2.
Pumping stations were placed along the pipeline where cumulative hydraulic head loss exceeded the maximum allowable pump head of 160 m. Head loss (H_{loss}) was computed as the sum of frictional and elevation-induced losses:

$$H_{loss} = H_{friction} + H_{elevation}$$

Frictional pressure drop was calculated using the Darcy-Weisbach equation:

$$H_{friction} = f \cdot \left(\frac{L}{D}\right) \cdot \left(\frac{v^2}{2g}\right)$$

Where f is the Darcy friction factor, L is pipe length, D is diameter, v is flow velocity, and g is gravitational acceleration. Friction factor f was determined using the Colebrook equation for turbulent flow:

$$\frac{1}{\sqrt{f}} = -2\, log10\left(\frac{\varepsilon}{3.7D} + \frac{2.51}{Re\sqrt{f}}\right)$$

Were ε was assumed to be 0.015 mm for GFRP[22].
Static head was calculated as:

$$H_{elevation} = max(\Delta z\,,\ 0)$$

Where Δz is the change in elevation over distance L. A pumping station was inserted whenever:

$$H_{loss} \geq H_{pump,max}$$

With $H_{pump,max}$ of 160 m per the chosen pump. This was an iterative calculation where cumulative losses were reset after each pumping station.

Table 2: Booster pumping station cost estimation.

Item	Unit Cost (US$)	Qty	Total Cost (US$)
Pump	30,000[24]	6	180,000
Pump isolation valves (DN650)	30,000	12	360,000
P-VRV	20,000[25]	1	20,000
VRV	10,000[25]	1	10,000
FI	15,000[26]	1	15,000
FCV (DN2000)	200,000[25]	1	200,000
Pipeline isolation valve (DN2000)	200,000[25]	1	200,000
Instrumentation & controls	20,000[23]	1	20,000
Building infrastructure	40,000	1	40,000

2.6 Risk modelling

2.6.1 Earthquakes

Sections of the pipeline crossing the mountain range between Hebei and Shanxi Prefectures crossed an area of high earthquake occurrence. As such it was important to estimate the probability of earthquakes occurring in the area which could affect the pipeline.

The Gutenberg-Righter Relationship can be written as:

$$\log\left(N\left(M\right)\right) = a - b * M \qquad (1)$$

Where N(M) describes the number of earthquakes of magnitude M or larger in a year and a, b being constant values determined by historical data for an area[27].
By determining the a and b constants for a small area using historical earthquake data, we can calculate the likelihood of earthquakes of a significant magnitude or higher or express that as a number of years on average before the next earthquake.

2.6.2 Pipe failure

Pipeline failure was modelled using a cumulative distribution function based on a failure rate adjusted by the length of the pipe as in equation (X):

$$F\left(t\right) = 1 - e^{\lambda Lt} \qquad (X)$$

Where λ is the failure rate in failures per year per kilometre, L is the pipe length in km and t is the time period in years. The time for maintenance could be calculated by assigning a threshold value of 0.95 for F(t) and determining the value of t, indicating the time interval before which the pipeline has a 5% chance of experiencing a rupture or break.

2.7 Method Limitations

1. NASA's DEM data is at a high spatial resolution (2.5 arcminutes[21]) and therefore requires some level of compression to optimise processing time.
2. R code runs on a single processor core with a high memory demand. Initial pathfinding runs took ~240 minutes to complete, this was progressively optimised down to ~3 minutes. It was also observed that computation was ~150-200% faster on an ARM CPU versus x86 architecture.
3. Bounding box was manually set based on elevation plots. Increasing bounding box size increased compute times exponentially. It was assumed that the optimal route would not extend beyond this bounding box. This is a reasonable assumption but may result in inconsistencies if repeated with larger bounding boxes.
4. Pathfinding cost function only accounted for distance and elevation. It was thus geographically agnostic, and the resultant path frequently cut through bodies of water and population centres. This was accepted as a reasonable error of a path of these lengths in a preliminary study stage. Further optimisation could be performed using land usage data[20] to avoid existing developments, as well as including multi-hazard maps which include seismic, climactic, and wildfire risk.
5. Modelling of risk is inherently probabilistic based on past data and testing and should appropriately be accounted for in the expectation of repair and maintenance.

3. **Results and Discussion**

3.1 Pipeline Routing

Two main routes with further sub-paths were chosen for comparison:

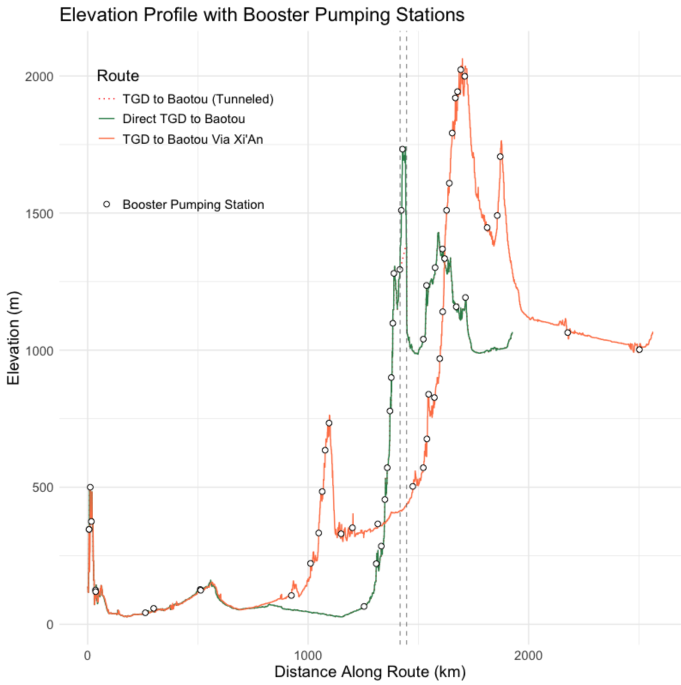


Figure 2:Change in frictional and elevational pressure drops along the pipeline.

1. Direct route: optimal path between the TGD and Baotou. This was further divided into:
   a. Direct route.
   b. Direct route with a tunnel across the highest peak.
2. Indirect route: TGD to Xi'An to Baotou. This aimed to determine whether the benefits of crossing the orographic divide early outweighed cost factors associated with suboptimal routing.

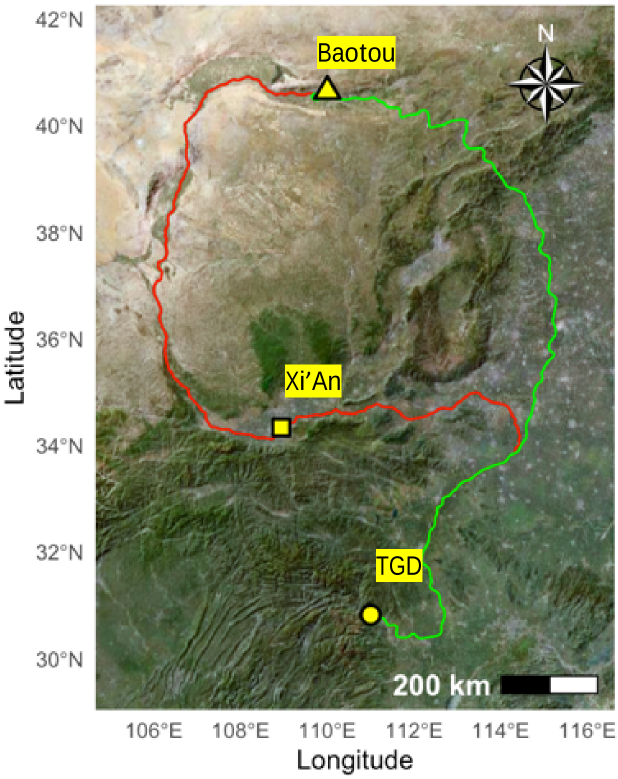


Figure 3: Proposed pipeline paths between TGD and Baotou. Direct route (green) and indirect route (red)

Route	Distance (km)	Total Elevation Gain (m)	Net Elevation Gain (m)	No. of booster pumping stations	Pumping station cost ($ US)	Operational cost ($ US p.a.)	Pumping power requirement (MW)
Direct route	1926	4359	1714	23	24,000,000	TBD	41

3.2 Hydraulic Modelling

Frictional pressure drop is proportional to pipeline length while static head is proportional to elevation gain.
Figure 2 indicates booster pumping station placement along each route based on the calculated pipeline length and elevation profiles.
The optimal route per pathfinding characteristics requires 23 booster pumping stations, while the indirect route via Xi'An requires 34 as per Table 3.
Inclusion of a tunnel reduces the length and therefore reduces the required pumping stations by 1 (22 required in total). A tunnel also reduces the pumping power requirement by 8 MW

Table 3: Comparison table between the chosen routes

(~20% decrease).

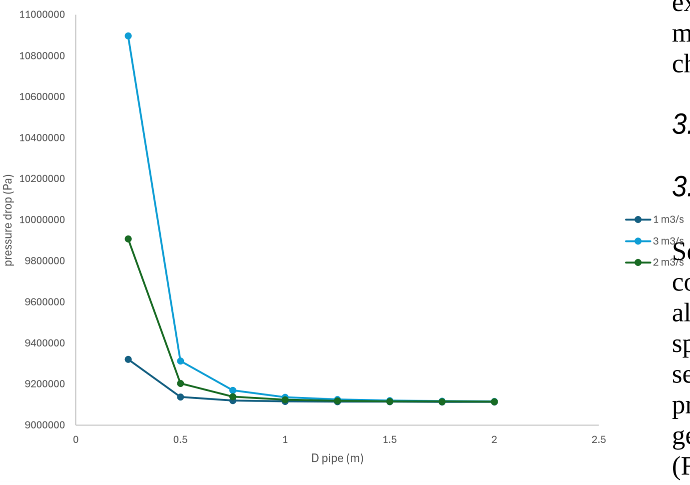


Figure 4: comparison of Pressure drops along the pipeline for varying flow rates and pipe diameters

Figure 4 demonstrates that across a range of acceptable volumetric flow rates of water, pipes of diameter 1 m or greater exhibit little difference in pressure drop while more constricted piping both always produces a larger pressure drop due to increased flow velocities but is also more greatly affected by changes in flow rate. A pipe width of 1 m corresponds to flow velocities of 1.3-4 m/s for flow rates 1-3 m^3/s which is an adequate agreement with common flow heuristics [28] for water transport.

Direct route with tunnel	1898	3942	1402	22	23,000,000	TBD	33
Indirect Route	2583	5391	2396	34	35,500,000	TBD	48

3.3 Consideration of tunnelling

Along the direct route between the Three Gorges Dam and Baotou there is a potential benefit to implementing a tunnel through a large mountain range to reduce energy expenditure on pumping associated with elevation increases. The rock type in this area has been identified as igneous rocks such as granite, dolerite and porphyry[29] which is difficult to bore through [30] indicating an increased CAPEX cost at the benefit of a reduced OPEX. Given the increased frequency of earthquakes in the area discussed section 3.4, the presence of 'harder' rock types does have the benefit of increased seismic activity resistance.
Projects have been completed to produce large tunnels through igneous terrain previously such as the Gotthard Base railway tunnel running through central Europe at a length of 85 km and cost of US$ 12 billion. At a similar rate it could be expected that a tunnel here would cost at a maximum US$ 6 billion for 40 km, but potentially cheaper due to Chinese labour values[31].

3.4 Climactic Risks and Impacts

3.4.1 Seasonal Water and Temperature Variation

Seasonal weather variations and region-specific conditions present a range of design challenges along the pipeline. For instance, the precipitation spikes engendered by the East Asian monsoon season must be accounted for, whereby the mean precipitation exceeds 100 mm per month in the general area between the TGD and Baotou (Figure. S1). Flooding is an issue for all geographical regions along the pipeline, with the Baotou area being more susceptible to intense flash flooding[32]. This induces the rise of hydrotechnical hazards including watercourse erosion, landslides, and vortex shedding[33]. These hazards are especially significant, as the proposed pipeline will be predominantly above ground. Not only can watercourse erosion and landslides undermine pipeline foundations, but they can also significantly increase turbidity which introduces an array of sediments to the intake water. Accordingly, water quality is reduced and internal stress increased. Moreover, there will be a surge in microbe quantity subsequent to flooding, in turn increasing the risk of corrosion and biofouling[34].

Considering the significant temperature variations throughout the year in China is imperative for optimal pipe design. Thermal expansion and contraction are inevitable due to the sizeable ambient temperature disparity between the summer season (> 30 ℃) and winter season (< -15 ℃) (Figure. S1). These repeated fluctuations can lead to cyclic fatigue, due to varying tensile and compressive stresses[35]. Additionally, regions near the TGD have a humid subtropical climate and experience the highest temperatures along the pipeline route (Figure. S2). Moisture, in tandem with high heat, exacerbates corrosion and promotes microbial growth. In addition, prolonged exposure to UV radiation can degrade not only coatings, but also the molecular chains in plastic pipes, resulting in embrittlement[36]. Conversely, geographical areas near Baotou reach temperatures well below freezing point (Figure. S2). When pipeline wall temperature is below freezing, ice formation on the inner surface could occur due to the heat temperature between the cold pipe and water. This can potentially cause problematic blockages, increase pressure drop, and reduce flow capacity[37].

3.4.2 Impacts on Pipeline Design

Filtration

A comprehensive filtration system is needed to mitigate the sedimentation impacts and to prevent risks from contamination. The pipeline filtration system should include the following:

Prefiltration: This initial stage is essential for the removal of larger particulate matter immediately following points. This is particularly crucial after extreme flooding events, where larger objects and particulate matter are displaced. Coarse bar screens made of stainless steel with spacings of 25 mm should be complemented by fine bar screens with spacings of 5 mm to capture large debris such as rags, branches, and plastics[38]. Additionally, hydrocyclones should also be positioned near intake points for further retention of suspended particles. Centrifugal force is utilised to remove particles (sand, silt, and other debris) to protect finer downstream filtration systems from potential damage and improve system efficiency[39]. Flocculation and settling basins will then be used to agglomerate and remove finer particles that remain post-cyclonic separation[40].

Secondary Filtration: This stage targets the removal of residual harmful particles and contaminants that have evaded primary treatment. The filtration units will be situated at intermediate stations, as well as near both intake and endpoint locations along the pipeline. This will ensure comprehensive contaminant removal throughout the pipeline network. Rapid sand filtration is employed to capture fine particles as small as 5 microns and is able to manage the specified flow rate of 2 m^3/s[41]. Additionally, activated carbon filtration is integrated to adsorb organic compounds, while ultrafiltration is utilized to eliminate microbial contaminants. This multifaceted approach is particularly crucial in humid regions (near the source) and in areas prone to extreme flooding, where the risk of organic matter accumulation and microbial proliferation is elevated. Subsequently, UV sterilisers will be used, as they are able to harness UV-C light to inactivate any remaining microorganisms by damaging their DNA[42].

Please note that further water clarification for uses such as consumption will be carried out at the site location.

Insulation and Coatings

To mitigate the harmful effects of the varying temperature conditions along the pipeline route, a segmented insulation strategy will be implemented. This will ensure that suitable insulation materials are chosen based on the specific thermal demands of disparate geographic areas, thereby optimising longevity and performance. In regions near Baotou subject to extremely cold winters, polyurethane foam will be used to prevent freezing due to its low thermal conductivity (~0.02 W/m·K) and ability to maintain its mechanical strength at subzero temperatures. This in turn minimises heat transfer, effectively enhancing energy efficiency and reducing operational costs[43]. However, polyurethane foam is susceptible to thermal degradation in higher temperatures, making it an unsuitable material for hotter climates. Conversely, for pipeline segments near the TGD and Central China with hotter summers and high humidity, calcium silicate insulation will be utilised. Calcium silicate high thermal stability and resistance to moisture absorption, making it ideal for subtropical climates. Furthermore, its robust structure prevents material breakdown under prolonged heat exposure, ensuring system reliability and long-term insulation efficiency.

Moreover, UV-resistant epoxy resin coatings will provide protection against solar radiation induced damage, corrosion, and chemical damage[44]. This should be applied in conjunction with a silver derived antimicrobial coating containing > 2000 mg/L silver zeolite to prevent the formation of a biofilm and impede biodeterioration[45].

Glass Fiber Reinforced Polymer piping (GFRP) is sensitive to both alkaline and acidic conditions. To monitor changes in water pH resulting from previously mentioned biomass and nutrient enrichment in the TGR, pH sensors should be installed near the intake points. The UV-resistant epoxy resin will act as a protective barrier to mitigate pH corrosion. Additionally, automated dosing systems will be implemented to release citric acid or calcium hydroxide when needed based on real time monitoring from the pH sensors.

Treatment Cost

Table 4. CAPEX and OPEX for filtration components, insulation materials, coating materials, and sensors.

	CAPEX (US$)	OPEX (US$/year)
Cost	15,000,000	1,700,000

The CAPEX and OPEX for the pipeline treatment system was calculated using published market values and established industry estimates. It was assumed that 22 units of coarse board screens, hydrocyclones, activated carbon, and UV-C sterilisers will be implemented at intake and outlet points, as well as near pumping stations. Furthermore, 2 units of settling basins and flocculation tanks was accounted for in the estimation[46,47].

3.4.3 Pipeline Sustainability

The pipeline location offers significant geographical advantages that can be harnessed for energy. Notably, the hydroelectric energy generated by the TGD and the abundant solar potential enable a substantial reduction in the operational carbon emissions of the pipeline, compared to the scenario where non-renewable energy sources are used (Figure 5).

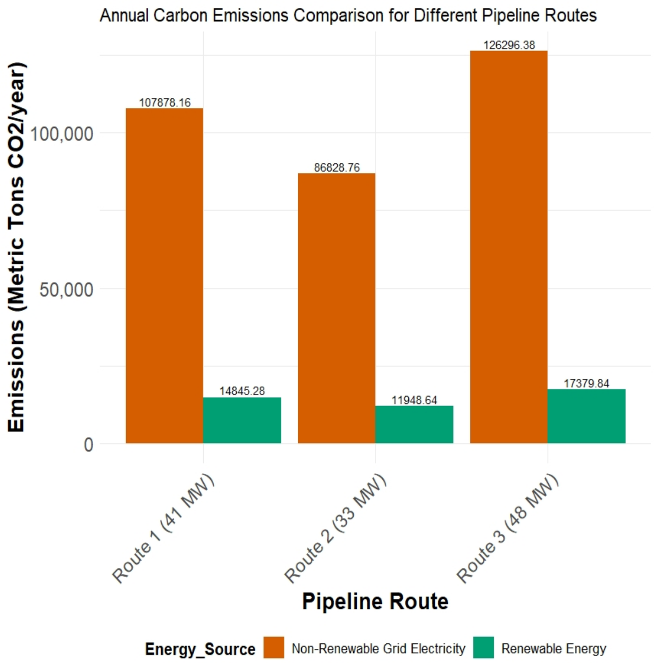


Figure 5. Carbon emissions produced for each pipeline route using non-renewable or renewable energy sources. Data was collected from the Australian Energy Market Operator[48].

3.5 Cost and Energy Optimisation

3.5.1 Pipeline CAPEX Estimation

Pipeline material cost was extrapolated using a standard HDPE piping chart from 32-800 mm diameter[49], resulting in an estimated cost of approximately US$ 760,000/km. Three CAPEX sheets were created for each proposed route: Route A: Tunnel option, Route B: Direct pumping over the mountain and Route C: Indirect route around terrain. A scaling factor of 1.5 was applied to the pipeline material cost to account for fittings, valves, and monitoring systems[50]. Additionally, the CAPEX for pumping stations was determined using data from Table 3. Construction costs were estimated at US$ 1.3 million/km, again based on pipeline length[50]. Extra construction costs were factored in for elevation and terrain access on the non-tunnel routes[51]. Tunnel construction was estimated at US$ 6 billion. A contingency of 10% was added to the final CAPEX figures.

3.5.2 Energy Optimisation and OPEX Analysis

Power supply was assumed to come from a mix of hydro and solar, both readily available along the pipeline route. It was assumed that a third of the pumping power demand would be met by the Three Gorges Hydroelectric Plant, the remaining two thirds would come from various solar farms positioned along the pipeline.

Purchasing energy from existing providers was determined to be more cost-effective than investing in proprietary generation infrastructure. However, battery systems were still considered essential for grid interruptions and remote operation. These were sized to provide 8 hours of backup supply. Lithium-ion battery costs in China were found to be US$ 88/kWh[52], and total battery costs were calculated per route based on specific energy requirements. Energy purchasing prices were: $ 42/MWh for hydro[53] and $ 49/MWh for solar[54]. With year-round operation assumed, the pipeline would require 8,760 MWh annually. Energy costs were thus calculated based on the pumping power requirement per route. Other OPEX considerations included maintenance: estimated at US$ 5,000/km annually[55], labour: assumed to be 2% of total pipeline cost, insurance: priced at 5% of the insured sum (total CAPEX), amortised over 50 years at a 3% annual interest rate[56] and environmental monitoring: assumed to be 40% of maintenance cost

3.5.3 Final Costing

Table 5. below shows the total cost for each route option based on both CAPEX and OPEX over a 50-year period. The results highlight that while the tunnelling route (Route A) saves on OPEX, its US$ 6 billion initial investment is a significant disadvantage. In contrast, the pumping route over the mountain (Route B) proves to be more cost-effective, despite its operational costs. Furthermore, potential risks associated with the tunnel, such as tunnel cave-ins and the high cost of repairs, also add a level of failure risk that must be considered when evaluating the long-term feasibility of the tunnel option.

Table 5. Total costs of each proposed route

Metric	**Value (Billion US$)**	**Total Cost (Billion US$)**
Total CAPEX (Option A)	13	23
Total OPEX (Option A)	10	
Total CAPEX (Option B)	8	20
Total OPEX (Option B)	12	
Total CAPEX (Option C)	11	27
Total OPEX (Option C)	16	

3.6 Failure Risk analysis

A large number of operational, financial and social risks are present in the construction and operation of the pipeline, including but not limited to: degradation of piping, joints, pumps and other structural elements; damage from natural disasters; errors in operation; costing overshoots and unexpected repair work; and impact on local communities.

In this section, a portion of the direct route pipeline covering the highest altitude section is analysed for several risk factors to provide insight into the risk considerations for a project of this size with further analysis left to later publication: damage and disruption from earthquakes as well as the financial risks of constructing a tunnel through the section.
Additionally, pipe failure is considered to determine maintenance scheduling requirements

Using the Gutenberg-Richter relationship to analyse the rough area over which the pipeline passes (bounded by latitudes 38-40 and longitudes 110-115) revealed that the average number of years before an earthquake of magnitude 4 or higher occurred affecting a given point was 42 years. Considering a project lifespan of 50 years, this is a significant frequency of earthquakes in the region. To mitigate the effects of earthquakes, Polyethylene or polymer piping such as Glass Fibre Reinforced Polymer (GFRP) should be used as it was found to be resistant to seismic activity in New Zealand[57], however a focus on fast repair responses should also be used as it is often impossible to design out the effects of an earthquake. Estimates for repairs to water pipelines damaged by seismic activity have been modelled to be in the range of US$ 10-20 million for lengths of 2.85 km pipe sections[58].

This location has also been identified as an optimal location for a tunnel given that there is an otherwise large increase and decrease in elevation that would require additional pumping power to overcome. Constructing a tunnel in this location poses a financial risk as it is a structure more prone to damage by an earthquake compared to a surface pipeline. As such in the evaluation of the cost effectiveness of the implementation of a tunnel, the cost to fix a collapsed tunnel is an important factor in the event of a serious earthquake. Extrapolating from historic costing for tunnel repairs (US$ 70 million to repair 2km of tunnel in the US[59]), a repair cost could be expected to be up to US$ 1.4 billion to repair the entire 40km length of tunnel, although the event that the entire tunnel would be that heavily damaged is unlikely to occur so as such a US$ 70 million cost of repair is more realistic and a reasonable cost for a low likelihood event.

Pipe failure is an important failure risk to consider as it can occur at any point on the pipeline and requires constant monitoring and frequent maintenance. There was very little available literature on the failure rate of GFRP however given the operating conditions not being high pressure, the failure rate of steel pipe was used instead as 0.00029 failures/yr/km [60]. Applying this calculation over the entire direct route yielded a maintenance schedule of every 1.1 months. This value is a reasonable repair and maintenance schedule and could be further expanded by the implementation of sensors to ensure efficient monitoring, maintenance, and safety. These include flow and pressure monitoring devices like electromagnetic and ultrasonic flow meters, which measure water flow without obstructing the pipeline, as well as pressure transmitters and differential pressure sensors that detect leaks or blockages[61].
Leak detection is critical, utilizing acoustic leak detectors, hydrocarbon/water-sensing fibre optic cables, and ground-penetrating radar (GPR) to identify potential failures without excavation. Structural integrity is monitored through strain gauges, distributed temperature sensors (DTS), and corrosion sensors, which help assess material stress, temperature variations, and corrosion risks [62].
Water quality is maintained using turbidity, pH, dissolved oxygen, and total organic carbon (TOC) sensors to detect contamination[63]. Additionally, remote sensing systems such as SCADA, satellite and IoT connectivity, and drone-based infrared imaging enable real-time monitoring, particularly in remote desert environments[63,64]. Given the harsh conditions of the Gobi Desert, these systems prioritize durability, remote connectivity, and energy efficiency, and could incorporate solar-powered sensors.

4. Conclusion

Three water pipeline routes were considered for the purpose of delivering 172.8 ML of water per day from the Three Gorges Reservoir to the city of Baotou in Northern China for use in combatting desertification as well as industrial uses. The results of this article were compiled in the decision matrix in appendix 1, resulting in the direct route without a tunnel being the optimal route. The Direct route without tunnel provides the greatest benefit in terms of construction and operational efficiency at a predicted cost of US$ 20 Billion across a 50 year lifespan. The route spans 1927 km running east of the Qinling mountain range before passing over the mountain range towards Baotou. Considerations of the varying climatic conditions across the route were made and it was found that thermal insulation for heat and cold protection were required to the sum of US$ 1.2 million and an annual cost of water treatment of US$ 15 million with an initial construction cost of US$ 1.7 million. Tunnelling was considered in the construction to minimise operational cost associated with pumping over mountains, however the CAPEX requirements were too large to justify this decision with the route costing a predicted additional US$ 3 billion. Risk factors associated with earthquakes and pipe failure were considered as examples of design considerations for the construction of this project with further risk factors for later publications.

Further research and modelling of this pipeline project should aim to quantify additional risk and profile stakeholders in the project such as industrial companies, farmers and citizens living along the pipeline to determine their stakeholder requirements.

Acknowledgements

Special Thanks to Dr. Gobinath Rajarathnam for his Guidance in the development of this report
We acknowledge the use of AI (ChatGPT and Microsoft Copilot) in the production and editing of code and guidance in researching for this project

Attributions

Graphical Abstract: Osama Rehman
Introduction: Osama Rehman
Pathfinding: Osama Rehman
Hydraulic Modelling: Osama Rehman and Matthew Wolfenden
Risk Modelling: Matthew Wolfenden
Climate analysis: Karinna Yee
Cost and energy optimisation: Jonathan Tapia

Other sections were worked on communally

References

1. Kassas, M. Desertification: a general review. *J Arid Environ* **30**, 115–128 (1995).

2. Cheng, L. *et al.* Estimation of the Costs of Desertification in China: A Critical Review. *Land Degrad Dev* **29**, 975–983 (2018).

3. China achieves 2024 growth target, bolsters global economy. https://english.www.gov.cn/news/202501/18/content_WS678ae501c6d0868f4e8eeef7.html.

4. How communities in China helped keep desertification at bay – DESERTIFICATION. https://desertification.wordpress.com/2019/09/06/how-communities-in-china-helped-keep-desertification-at-bay/.

5. China's 'Great Green Wall' Fights Expanding Gobi Desert. https://www.nationalgeographic.com/science/article/china-great-green-wall-gobi-tengger-desertification.

6. Lyu, Y. *et al.* Desertification Control Practices in China. *Sustainability 2020, Vol. 12, Page 3258* **12**, 3258 (2020).

7. China completes 3,000-km green belt around its biggest desert, state media says | Reuters. https://www.reuters.com/world/china/china-completes-3000-km-green-belt-around-its-biggest-desert-state-media-says-2024-11-29/.

8. Wang, X., Chen, F., Hasi, E. & Li, J. Desertification in China: An assessment. *Earth Sci Rev* **88**, 188–206 (2008).

9. World Bank Group. Halting Desertification in China. https://www.worldbank.org/en/results/2021/07/26/halting-desertification-in-china (2021).

10. Sun, S., Xiang, W., Ouyang, S., Hu, Y. & Peng, C. Balancing Water Yield and Water Use Efficiency Between Planted and Natural Forests: A Global Analysis. *Glob Chang Biol* **30**, (2024).

11. Lu, C., Zhao, T., Shi, X. & Cao, S. Ecological restoration by afforestation may increase groundwater depth and create potentially large ecological and water opportunity costs in arid and semiarid China. *J Clean Prod* **176**, 1213–1222 (2018).

12. Nkonya, E., Mirzabaev, A. & von Braun, J. Economics of land degradation and improvement - A global assessment for sustainable development. *Economics of Land Degradation and Improvement - A Global Assessment for Sustainable Development* 1–686 (2015) doi:10.1007/978-3-319-19168-3/COVER.

13. Zhu, B. Q., Zhang, J. X. & Sun, C. Potential links of gobi, dust, and desertification: A comprehensive understanding from aeolian landform evolution in a middle-latitude desert. *Sediment Geol* **428**, 106049 (2022).

14. Ministry of Water Resources. *Implementing Water-Related Goals of the United Nations 2030 Agenda For Sustainable Development Chapter 3*. (2023).

15. Wang, Q. Q. *et al.* Water Conservation and Ecological Water Requirement Prediction of Mining Area in Arid Region Based on RS-GIS and InVEST: A Case Study of Bayan Obo Mine in Baotou, China. *Sustainability (Switzerland)* **15**, 4238 (2023).

16. Zeng, X. *et al.* Drainage-divide migration at the Qinling and its implications for the drainage reorganization and rift evolution of the Weihe graben, Central China. *Geomorphology* **475**, 109658 (2025).

17. Zhu, Z. N. *et al.* Monitoring of Yangtze River Discharge at Datong Hydrometric Station Using Acoustic Tomography Technology. *Front Earth Sci (Lausanne)* **9**, 723123 (2021).

18. Li, Z. *et al.* Water quality trends in the Three Gorges Reservoir region before and after impoundment (1992–2016). *Ecohydrology & Hydrobiology* **19**, 317–327 (2019).

19. Bruschi, R. From the Longest to the Deepest Pipelines. Preprint at https://dx.doi.org/ (2012).

20. Global Urban Footprint. https://www.dlr.de/en/eoc/research-transfer/projects-missions/global-urban-footprint.

21. Berry, P. A. M., Smith, R. G. & Benveniste, J. ACE2: The New Global Digital Elevation Model. *International Association of Geodesy Symposia* **135**, 231–237 (2010).

22. Inline Centrifugal Water Pump Manufacturers, Factory Price - JUSHI PUMP. https://www.jushipump.com/product/horizontal-chemical-process-pipeline-inline-centrifugal-water-pump/.

23. DEE, G. M. Jones. & Robert L. Sanks, P. PE. Pumping Station Design. (2011).

24. DELIVERING PUMPING SOLUTIONS 2020 PRICE LIST.

25. FLOMATIC Valves 2025 Price List. https://www.flomatic.com/wp-content/uploads/2025/02/Price-List-2025.pdf.

26. SITRANS FS230 - Siemens Global. https://www.siemens.com/global/en/products/automation/process-instrumentation/flow-measurement/ultrasonic/clamp-on/sitrans-fs230.html.

27. Southern California Earthquake Center. Gutenberg-Richter Relationship.

28. Couper, J. R., Penney, W. R., Fair, J. R. & Walas, S. M. Rules of Thumb: Summary. *Chemical Process Equipment* xiii–xx (2012) doi:10.1016/B978-0-12-396959-0.00031-8.

29. Li, Z. *et al.* Age and petrogenesis of ore-bearing porphyry from the Houyu Mo-polymetallic deposit, central North China Craton. *Ore Geol Rev* **169**, (2024).

30. Xu, H., Geng, Q., Sun, Z. & Qi, Z. Full-scale granite cutting experiments using tunnel boring machine disc cutters at different free-face conditions. *Tunnelling and Underground Space Technology* **108**, 103719 (2021).

31. Associated Press. After 17 years and $12 billion, Switzerland inaugurates world's longest rail tunnel. *Los Angeles Times* (2016).

32. China floods: The families torn apart by 'huge, furious waves'. https://www.bbc.com/news/world-asia-china-66458546.

33. Dooley, C., Prestie, Z., Ferris, G., Fitch, M. & Zhang, H. Approaches for evaluating the vulnerability of pipelines at water crossings. *Proceedings of the Biennial International Pipeline Conference, IPC* **2**, (2014).

34. Dzodzomenyo, M., Asamoah, M., Li, C., Kichana, E. & Wright, J. Impact of flooding on microbiological contamination of domestic water sources: a longitudinal study in northern Ghana. *Appl Water Sci* **12**, 1–10 (2022).

35. PIPA POP010A-Part 1-Polyethylene Pressure Pipes Design for Dynamic Stresses-Issue Part 1: Polyethylene Pressure Pipes Design for Dynamic Stresses.

36. Pipe Institute, P. TR-18/2019. (2019).

37. Kitanin, L., Smirnov, Y. A. & Lebedev, M. E. Development of Flow and Heat Transfer During Filling a Pipeline with Water at the Pipe Wall Temperature Below the Freezing Point. *Journal of Engineering Physics and Thermophysics* **89**, 808–814 (2016).

38. Bar Screen In Wastewater Treatment - Water & Wastewater. https://www.waterandwastewater.com/bar-screen-in-wastewater-treatment/.

39. Soccol, O. J. & Botrel, T. A. Hidrociclone para pré-filtragem da água de irrigação. *Sci Agric* **61**, 134–140 (2004).

40. Terry, B. L. COAGULATION, FLOCCULATION AND CLARIFICATION OF DRINKING WATER Engelhardt, Application Development Manager, Drinking Water. (2014).

41. Rapid Sand Filtration: Efficient Water Treatment Method - Water & Wastewater. https://www.waterandwastewater.com/rapid-sand-filtration-efficient-water-treatment-method/.

42. Chen, J. *et al.* Self-powered antifouling UVC pipeline sterilizer driven by the discharge stimuli based on the modified freestanding rotary triboelectric nanogenerator. *Nano Energy* **95**, 106969 (2022).

43. Physical properties of polyurethane insulation Safe and sustainable construction with polymers.

44. Tuominen, O., Tuominen, E., Vainio, M., Ruuska, T. & Vinha, J. Thermal and moisture properties of calcium silicate insulation boards. *MATEC Web of Conferences* **282**, 02065 (2019).

45. Vela-Cano, M., Garcia-Fontana, C., Osorio, F., González-Martinez, A. & González-López, J. Silver-Derived Antimicrobial Coatings for the Prevention of Microbial Biofilms in Metal Pipes. *Water Air Soil Pollut* **232**, 1–11 (2021).

46. Towler, G. *et al.* CHEMICAL ENGINEERING DESIGN Principles, Practice and Economics of Plant and Process Design. (2008).

47. Sper Scientific Instruments | Environmental Measurement Instruments – Sper Scientific Direct. https://sperdirect.com/.

48. AEMO | Australian Energy Market Operator. https://aemo.com.au/.

49. HDPE Pipe Prices & Sizes Made Amazingly Easy by Matrix Piping! https://www.matrixpiping.com.au/pages/poly-pipe-prices.

50. Anglo Coal Australia Pty Ltd. *Report for Dawson Valley Pipeline Cost Estimate*. https://www.aer.gov.au/system/files/GHD%20report%2C%205%20February%202005%20%28public%20copy%29.pdf (2006).

51. Case Study Oven Mountain Pump Hydro Project Australian Industry Plan — Hughes et al — Hughes et al. https://www.hughesetal.com.au/case-study-oven-mountain-pump-hydro-project-australian-industry-plan.

52. Lithium-ion battery pack prices fall 20% in 2024. https://www.energy-storage.news/lithium-ion-battery-pack-prices-fall-20-in-2024-amidst-fight-for-market-share/.

53. 中国长江三峡工程开发总公司. https://web.archive.org/web/20090210072449/http://www.ctgpc.com.cn/sx/news.php?mNewsId=29096.

54. IRENA. Renewable Power Generation Costs In 2022 - Executive Summary. *International Renewable Energy Agency* 69 (2023).

55. Cost of Corrosion - Gas & Liquid Transmission Pipelines - Rust Bullet, LLC. https://www.rustbullet.com/cost-of-corrosion/advantage-gas-and-liquid-transmission/.

56. Gladstone Area Water Board. (2024).

57. O'callaghan, F. W. PIPELINE PERFORMANCE EXPERIENCES DURING SEISMIC EVENTS IN NEW ZEALAND, 1987 TO 2015.

58. Mazumder, R. K., Fan, X., Salman, A. M., Li, Y. & Yu, X. Framework for Seismic Damage and Renewal Cost Analysis of Buried Water Pipelines. *J Pipeline Syst Eng Pract* **11**, 04020038 (2020).

59. Update on the 2019 Tunnel Collapse and Canal Washout – Tunnel Repair or Replacement Options | IANR News. https://ianrnews.unl.edu/update-2019-tunnel-collapse-and-canal-washout-tunnel-repair-or-replacement-options?utm_source=chatgpt.com.

60. Muhlbauer, W. K. Absolute Risk Estimates. *Pipeline Risk Management Manual* 293–329 (2004) doi:10.1016/B978-075067579-6/50017-0.

61. Ayadi, A., Ghorbel, O., BenSalah, M. S. & Abid, M. A framework of monitoring water pipeline techniques based on sensors technologies. *Journal of King Saud University - Computer and Information Sciences* **34**, 47–57 (2022).

62. Fan, H., Tariq, S. & Zayed, T. Acoustic leak detection approaches for water pipelines. *Autom Constr* **138**, 104226 (2022).

63. Abbasi, Z., Niazi, H., Abdolrazzaghi, M., Chen, W. & Daneshmand, M. Monitoring pH Level Using High-Resolution Microwave Sensor for Mitigation of Stress Corrosion Cracking in Steel Pipelines. *IEEE Sens J* **20**, 7033–7043 (2020).

64. Fonseca, C. A. A. SCADA System of Pipelines. *Handbook of Pipeline Engineering* 1–28 (2024) doi:10.1007/978-3-031-05735-9_18-1.

65. Green, D. W. & Perry, R. H. *Perry's Chemical Engineers' Handbook, Eighth Edition*. *Perry's Chemical Engineers' Handbook* (McGraw-Hill Education, 2008).

66. Temperature and precipitation gridded data for global and regional domains derived from in-situ and satellite observations. https://cds.climate.copernicus.eu/datasets/insitu-gridded-observations-global-and-regional?tab=overview.

Appendix

Appendix 1: Decision matrix on selection of route

Route		WEIGHTING		1 (direct with tunnel)		2 (direct, no tunnel)		3 (Indirect via Xi'An)	
				value	weighted	value	weighted	value	weighted
OPEX		10		3	30	2	20	1	10
CAPEX		8		1	8	3	24	2	16
RISKS									
climate		5		2	10	2	10	2	10
seismic		2		1	2	2	4	3	6
monetary		3		1	3	3	9	2	6
PPP									
People		5		1	5	1	5	2	10
Profits		2		1	2	1	2	2	4
Planet		5		2	10	3	15	1	5
total		40			1.75		2.225		1.675

Appendix 2: LLM Prompt Philosophy

A combination of ChatGPT (utilising the GPT-4o engine) and Microsoft CoPilot were utilised to generate R code. Prompting was performed by requesting the LLM to utilise specific known equations from reputable references (e.g. Darcy's Law from Perry's Handbook[65].

Code was generated in short sections, tested, and compiled. For example,

1. Generate code to import and process DEM data.
2. Generate code to perform pathfinding between two latitude and longitude co-ordinates – output results on a DEM plot.
3. Modify the previous code to have a cost function between elevation and distance (where elevation costs x times as much as distance).
4. Modify the code to output the route over satellite imagery utilising a known GEOTIFF (map.tiff).

Each iteration was tested and debugged.

Appendix 3: Climactic Modelling

R code and figure for Fig. S1.

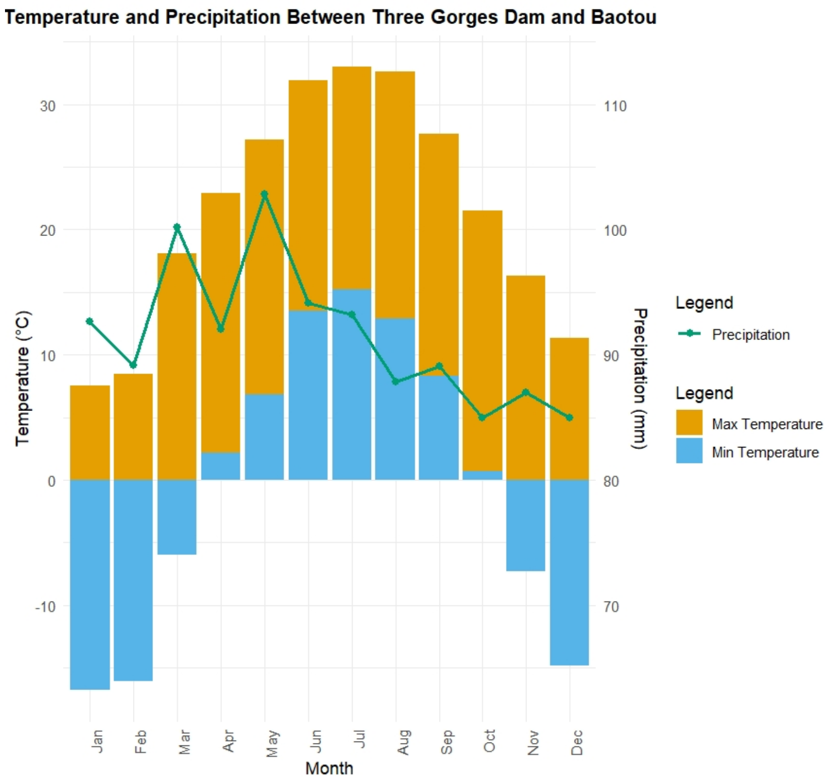


Fig. S1. The minimum and maximum temperature and precipitation for the general region between the TGD and Baotou. The data was derived from The Climate Data Store - "Temperature and precipitation data gridded data for global and regional domains derived from in-situ and satellite observations"[66].

```
library(ncdf4)

library(ggplot2)

library(dplyr)
```



```
max_temp_file_path <- "C:/Users/keith/Documents/max_temp.nc"

min_temp_file_path <- "C:/Users/keith/Documents/min_temp.nc"

nc_max_data <- nc_open(max_temp_file_path)

nc_min_data <- nc_open(min_temp_file_path)

lon <- ncvar_get(nc_max_data, "lon")

lat <- ncvar_get(nc_max_data, "lat")

time <- ncvar_get(nc_max_data, "time")

max_temperature <- ncvar_get(nc_max_data, "tasmax")

min_temperature <- ncvar_get(nc_min_data, "tasmin")

time <- as.Date(time, origin = "1970-01-01")
three_gorges_coords <- c(30.8231, 111.0031)

baotou_coords <- c(40.6562, 109.8345)

find_nearest_index <- function(array, value) {

  which.min(abs(array - value)) }

three_gorges_lat_idx <- find_nearest_index(lat, three_gorges_coords[1])

three_gorges_lon_idx <- find_nearest_index(lon, three_gorges_coords[2])

baotou_lat_idx <- find_nearest_index(lat, baotou_coords[1])

baotou_lon_idx <- find_nearest_index(lon, baotou_coords[2])

area_max_temp_data <- max_temperature[three_gorges_lon_idx:baotou_lon_idx, three_gorges_lat_idx:baotou_lat_idx, ]

area_min_temp_data <- min_temperature[three_gorges_lon_idx:baotou_lon_idx, three_gorges_lat_idx:baotou_lat_idx, ]

max_temp_per_time <- apply(area_max_temp_data, 3, max, na.rm = TRUE)

min_temp_per_time <- apply(area_min_temp_data, 3, min, na.rm = TRUE)

max_temp_df <- data.frame(

  time = time,

  temperature = max_temp_per_time,

  legend = "Max"

)

min_temp_df <- data.frame(

  time = time,
```



```
    temperature = min_temp_per_time,

    legend = "Min"

)

temp_df <- bind_rows(max_temp_df, min_temp_df)

temp_df <- temp_df %>%

    mutate(month = format(time, "%Y-%m")) %>%

    group_by(month, legend) %>%

    summarize(temperature = if_else(legend == "Max", max(temperature, na.rm = TRUE), min(temperature, na.rm = TRUE)))

precipitation_data <- data.frame(

    month = c("Jan", "Feb", "Mar", "Apr", "May", "Jun", "Jul", "Aug", "Sep", "Oct", "Nov", "Dec"),

    precipitation = c(92.69148, 89.14843, 100.20638, 92.00117, 102.80787, 94.14516, 93.18654, 87.84467, 89.05898,
84.96056, 87, 85)

)

temp_df <- temp_df %>%

    mutate(month_name = format(as.Date(paste0(month, "-01")), "%b"))

precipitation_data <- precipitation_data %>%

    mutate(month_name = factor(month, levels = month.abb))

temp_df$month_name <- factor(temp_df$month_name, levels = month.abb)
ggplot() +

    geom_bar(data = temp_df %>% filter(legend == "Max"), aes(x = month_name, y = temperature), stat = "identity", fill =
"#E69F00", position = "dodge") +

    geom_bar(data = temp_df %>% filter(legend == "Min"), aes(x = month_name, y = temperature), stat = "identity", fill =
"#56B4E9", position = position_dodge(width=0.9)) +

    geom_line(data = precipitation_data, aes(x = month_name, y = precipitation - 80, group = 1), color = "#009E73", size = 1)
+

    geom_point(data = precipitation_data, aes(x = month_name, y = precipitation - 80), color = "#009E73", size = 2) +

    scale_y_continuous(

        name = "Temperature (°C)",

        sec.axis = sec_axis(~ . + 80, name = "Precipitation (mm)")

    ) +

    labs(title = "Temperature and Precipitation Between Three Gorges Dam and Baotou",
```



```
        x = "Month",
        fill = "Legend",
        color = "Legend") +
    theme_minimal() +
    theme(axis.text.x = element_text(angle = 90, hjust = 1),
          plot.title = element_text(size = 12, hjust = 0.5, face = "bold"))
nc_close(nc_max_data)
nc_close(nc_min_data)
```


R code and figure for Fig. S2.

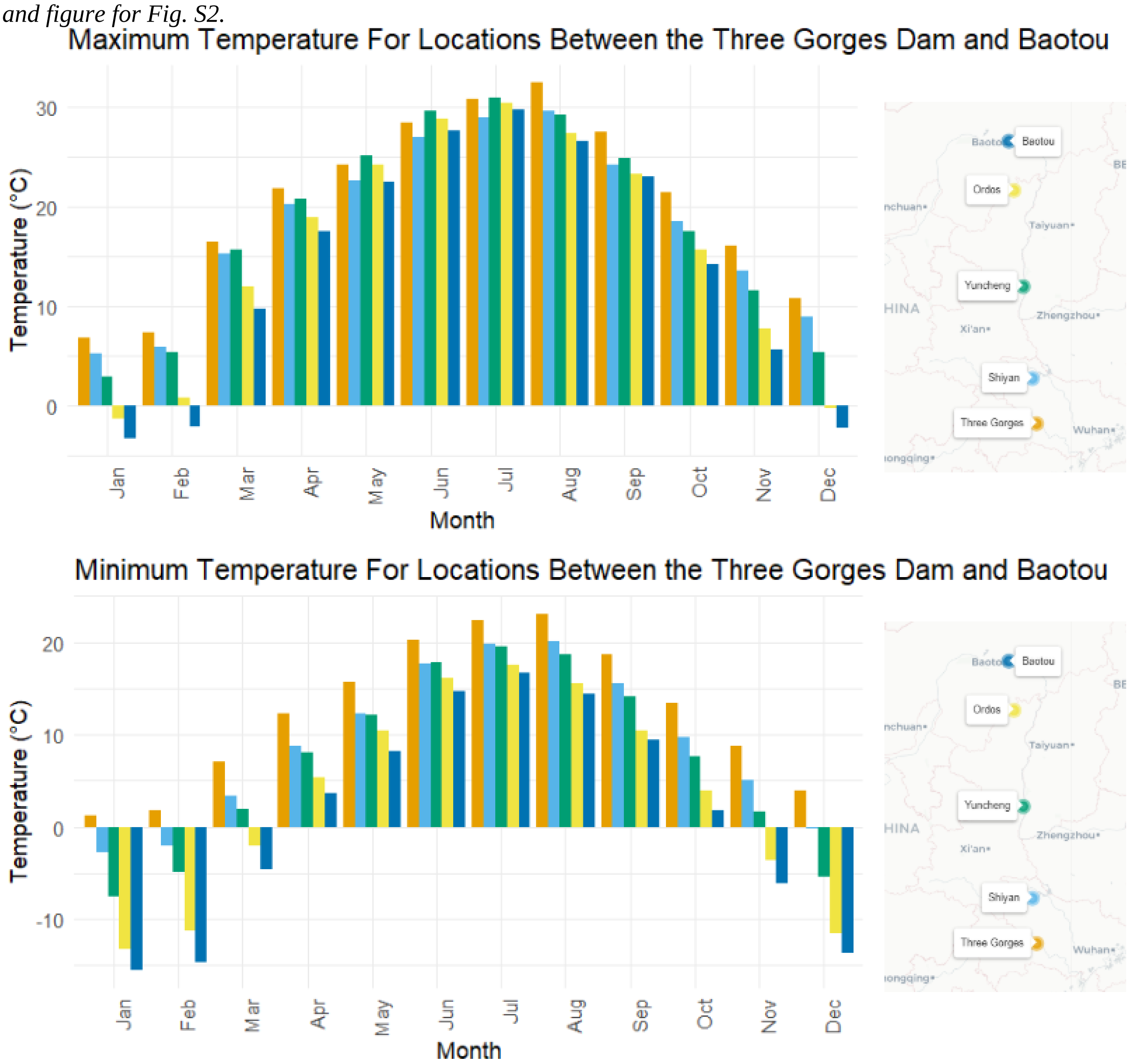


Fig. S2. Minimum and maximum temperatures for disparate nodes between the TGD and Baotou. The data was derived from The Climate Data Store - "Temperature and precipitation data gridded data for global and regional domains derived from in-situ and satellite observations"[66].

```
library(ncdf4)

library(ggplot2)

library(dplyr)

library(gridExtra)

library(ggthemes)
```



```
max_temp_file_path <- "C:/Users/keith/Documents/max_temp.nc"

min_temp_file_path <- "C:/Users/keith/Documents/min_temp.nc"

nc_max_data <- nc_open(max_temp_file_path)

nc_min_data <- nc_open(min_temp_file_path)

lon <- ncvar_get(nc_max_data, "lon")

lat <- ncvar_get(nc_max_data, "lat")

time <- ncvar_get(nc_max_data, "time")

max_temperature <- ncvar_get(nc_max_data, "tasmax")

min_temperature <- ncvar_get(nc_min_data, "tasmin")

time <- as.Date(time, origin = "1970-01-01")

coordinates <- list( c(30.8231, 111.0031), # Three Gorges c(32.46195, 110.8083), # Shiyan c(35.73965, 110.4188), # Yuncheng c(39.01735, 110.0293), # Ordos c(40.6562, 109.8345) # Baotou ) city_names <- c("Three Gorges", "Shiyan", "Yuncheng", "Ordos", "Baotou")

find_nearest_index <- function(array, value) { which.min(abs(array - value)) }

extract_temp_data <- function(coords, temperature) { temp_data <- list() for (coord in coords) { lat_idx <- find_nearest_index(lat, coord[1]) lon_idx <- find_nearest_index(lon, coord[2]) temp_data[[paste(coord[1], coord[2], sep = ", ")]] <- temperature[lon_idx, lat_idx, ] } return(temp_data) }

max_temp_data <- extract_temp_data(coordinates, max_temperature)

min_temp_data <- extract_temp_data(coordinates, min_temperature)

calculate_monthly_temp <- function(temp_data, time, type, city_names) { temp_df_list <- list() for (i in seq_along(temp_data)) { name <- names(temp_data)[i] city <- city_names[i] temp_df <- data.frame( time = time, temperature = temp_data[[name]], legend = paste(type, city) ) temp_df <- temp_df %>% mutate(month = format(time, "%Y-%m")) %>% group_by(month, legend) %>% summarize(temperature = if (type == "Max") max(temperature, na.rm = TRUE) else min(temperature, na.rm = TRUE)) temp_df_list[[name]] <- temp_df } return(do.call(rbind, temp_df_list)) }

max_temp_df <- calculate_monthly_temp(max_temp_data, time, "Max", city_names)

min_temp_df <- calculate_monthly_temp(min_temp_data, time, "Min", city_names)

temp_df_max <- max_temp_df

temp_df_min <- min_temp_df

temp_df_max <- temp_df_max %>% mutate(month_name = format(as.Date(paste0(month, "-01")), "%b")) temp_df_min <- temp_df_min %>% mutate(month_name = format(as.Date(paste0(month, "-01")), "%b"))

temp_df_max$month_name <- factor(temp_df_max$month_name, levels = month.abb) temp_df_min$month_name <- factor(temp_df_min$month_name, levels = month.abb)
```



```
city_order <- c("Three Gorges", "Shiyan", "Yuncheng", "Ordos", "Baotou")

temp_df_max$legend <- factor(temp_df_max$legend, levels = paste("Max", city_order))

temp_df_min$legend <- factor(temp_df_min$legend, levels = paste("Min", city_order))

colorblind_palette <- c( "#E69F00", # Orange "#56B4E9", # Sky Blue "#009E73", # Bluish Green "#F0E442", # Yellow
"#D55E00" # Vermillion )

p1 <- ggplot(temp_df_max, aes(x = month_name, y = temperature, fill = legend)) + geom_bar(stat = "identity", position =
"dodge") + labs(title = "Maximum Temperature For Locations Between the Three Gorges Dam and Baotou", x = "Month", y
= "Temperature (°C)", fill = "Legend") + scale_fill_manual(values = colorblind_palette) + # Use colorblind-friendly colors
theme_minimal() + theme(axis.text.x = element_text(angle = 90, hjust = 1))

p2 <- ggplot(temp_df_min, aes(x = month_name, y = temperature, fill = legend)) + geom_bar(stat = "identity", position =
"dodge") + labs(title = "Minimum Temperature For Locations Between the Three Gorges Dam and Baotou", x = "Month", y
= "Temperature (°C)", fill = "Legend") + scale_fill_manual(values = colorblind_palette) + # Use colorblind-friendly colors
theme_minimal() + theme(axis.text.x = element_text(angle = 90, hjust = 1))

grid.arrange(p1, p2, ncol = 1)

library(leaflet)

locations <- data.frame(
  Name = c("Three Gorges", "Shiyan", "Yuncheng", "Ordos", "Baotou"),
  Latitude = c(30.8231, 32.46195, 35.73965, 39.01735, 40.6562),
  Longitude = c(111.0031, 110.8083, 110.4188, 110.0293, 109.8345),
  Color = c("#FF0000", "#808000", "#008080", "#0000FF", "#800080")
)
map <- leaflet(data = locations) %>%
  addProviderTiles(providers$CartoDB.Positron) %>%
  setView(lng = 110, lat = 35, zoom = 5)
map <- map %>%
  addCircleMarkers(
    lng = ~Longitude,
    lat = ~Latitude,
    color = ~Color,
    label = ~Name,
    labelOptions = labelOptions(noHide = TRUE, direction = 'auto'),
    radius = 5,
    fillOpacity = 0.8
  )
Map
```


Appendix 4: Pathfinding Code

The following code utilises NASA DEM data[21] and computes the optimal path between two lat / long co-ordinates. Output is a .GPX route file (to be processed in a different R script).

```
gc()

# Load required libraries
library(terra)
library(gdistance)
library(sf)
library(pbapply)
library(raster)  # Added raster package for conversion
library(ggplot2)
library(viridis)

# Define node coordinates
nodes <- data.frame(
  name = c("Three Gorges Dam", "Baotou"),
  lon = c(111.0037, 109.8402),
  lat = c(30.8233, 40.6578)
)

# Load or generate the reduced DEM file
# dem_path <- "Reduced_DEM.tif"
dem_path <- "Reduced_DEM.tif"
start_time_total <- Sys.time()  # Start total time tracking

start_time_dem_generation <- Sys.time()  # Start time for reduced DEM generation
if (!file.exists(dem_path)) {
  cat("Reduced DEM not found. Generating it from original DEM...\n")
  original_path <- "Merged_China_DEM.tif"
  if (!file.exists(original_path)) {
    stop("Error: Original DEM file not found at", original_path)
  }
  original_dem <- rast(original_path)
  reduced_dem <- aggregate(original_dem, fact = 4, fun = mean, na.rm = TRUE)  # Reduce
resolution
  writeRaster(reduced_dem, dem_path, overwrite = TRUE)
  cat("Reduced DEM saved as", dem_path, "\n")
}
end_time_dem_generation <- Sys.time()
cat("Time taken for reduced DEM generation: ", difftime(end_time_dem_generation,
start_time_dem_generation, units = "mins"), "\n")

# Load the reduced DEM
cat("Loading DEM data...\n")
dem_raster <- rast(dem_path)

# Define bounding box coordinates for the new region
min_lon <- 106  # Minimum longitude (105°E)
max_lon <- 116  # Maximum longitude (120°E)
min_lat <- 30   # Minimum latitude (25°N)
max_lat <- 42   # Maximum latitude (45°N)

# Set bounding box for elevation data
```



```
cat("Setting bounding box from 105°E to 120°E and 25°N to 45°N...\n")
dem_bbox <- c(xmin = min_lon, xmax = max_lon, ymin = min_lat, ymax = max_lat)

# Crop the DEM to the bounding box region
cat("Cropping DEM to the new bounding box...\n")
dem_raster <- crop(dem_raster, dem_bbox)

# Check for NA values
cat("Checking for NA values in DEM...\n")
dem_raster[is.na(dem_raster)] <- max(values(dem_raster), na.rm = TRUE)  # Assign high
cost to NA

# Set cache directory for performance improvement
cache_dir <- "F:/DEMs/cache"
if (!dir.exists(cache_dir)) dir.create(cache_dir, recursive = TRUE)
terraOptions(tempdir = cache_dir)

# Convert DEM to RasterLayer for compatibility with gdistance
cat("Converting SpatRaster to RasterLayer for transition matrix computation...\n")
dem_raster_layer <- raster(dem_raster)

# Convert DEM to transition matrix with adjusted cost for elevation
cat("Computing transition matrix with adjusted cost for elevation...\n")
start_time_transition <- Sys.time()  # Start time for transition matrix computation
progress_bar <- txtProgressBar(min = 0, max = 1, style = 3)

tryCatch({
  cost_surface <- transition(dem_raster_layer, transitionFunction = function(x) {
    # Calculate the average elevation value
    val <- mean(x, na.rm = TRUE)

    # Calculate the elevation change (difference) and distance
    elevation_change <- max(x, na.rm = TRUE) - min(x, na.rm = TRUE)
    distance <- sqrt(sum(diff(c(x[1], x[2]))^2))  # Euclidean distance between two points

    # Apply the weight factor for elevation change
    elevation_cost <- 0.00015 * elevation_change  # Elevation cost is 10 times the distance

    # Combine the distance cost and elevation cost
    total_cost <- distance + elevation_cost

    # Avoid negative or zero values
    if (total_cost > 0) {
      return(1 / total_cost)  # Inverse of total cost for transition matrix
    } else {
      return(Inf)  # Prevent negative or zero values
    }
  }, directions = 8, symm = FALSE)
  cost_surface <- geoCorrection(cost_surface, type = "c", multpl = FALSE)
  setTxtProgressBar(progress_bar, 1)
}, error = function(e) {
  stop("Memory error during transition matrix computation: ", e$message)
})

close(progress_bar)
end_time_transition <- Sys.time()
cat("Time taken for transition matrix computation: ", difftime(end_time_transition,
start_time_transition, units = "mins"), "\n")
```

```
# Convert nodes to spatial points
cat("Processing nodes...\n")
node_points <- st_as_sf(nodes, coords = c("lon", "lat"), crs = 4326)
node_points <- st_transform(node_points, crs = st_crs(dem_raster))
node_points <- st_geometry(node_points)  # Ensure geometry consistency

# Function to smooth path coordinates using smooth.spline
smooth_path <- function(path_sf) {
  coords <- st_coordinates(path_sf)
  smoothed_coords <- data.frame(
    lon = smooth.spline(coords[, 1])$y,  # Smooth the longitude
    lat = smooth.spline(coords[, 2])$y   # Smooth the latitude
  )

  # Create a new Simple Feature object with smoothed coordinates
  smoothed_path <- st_as_sf(st_sfc(st_linestring(as.matrix(smoothed_coords)), crs =
st_crs(path_sf)))
  return(smoothed_path)
}

# Initialize variable to accumulate total distance
total_distance <- 0

# Function to compute path between two nodes sequentially and print path distance
compute_path <- function(i) {
  cat("Computing path for", nodes$name[i], "to", nodes$name[i+1], "...\n")
  start_time_path <- Sys.time()  # Start time for path computation
  tryCatch({
    path <- shortestPath(cost_surface, as.numeric(st_coordinates(node_points[i])),
                         as.numeric(st_coordinates(node_points[i+1])), output = "SpatialLines")

    # Convert path to sf object
    path_sf <- st_as_sf(st_sfc(st_as_sfc(path), crs = st_crs(dem_raster)))

    # Smooth the path
    smoothed_path_sf <- smooth_path(path_sf)

    # Calculate total path length
    path_length <- st_length(smoothed_path_sf)  # Length of the path (in meters, depending on
CRS)
    cat("Total path length from", nodes$name[i], "to", nodes$name[i+1], ":",
round(path_length, 2), "meters\n")

    # Accumulate the path length in total_distance
    total_distance <<- total_distance + as.numeric(path_length)  # Add the path length to the
total distance

    return(smoothed_path_sf)  # Return the smoothed path
  }, error = function(e) {
    cat("Error computing path:", e$message, "\n")
    return(NULL)
  })
  end_time_path <- Sys.time()
  cat("Time taken for path from", nodes$name[i], "to", nodes$name[i+1], ":",
difftime(end_time_path, start_time_path, units = "secs"), "\n")
}
```



```
# Compute paths sequentially with progress bar
cat("Computing paths sequentially to reduce RAM usage...\n")
progress_bar <- txtProgressBar(min = 0, max = nrow(nodes) - 1, style = 3)
paths <- vector("list", length = nrow(nodes) - 1)
for (i in 1:(nrow(nodes) - 1)) {
  paths[[i]] <- compute_path(i)
  setTxtProgressBar(progress_bar, i)
}
close(progress_bar)


# Save paths
cat("Saving computed paths...\n")
valid_paths <- paths[!sapply(paths, is.null)]

if (length(valid_paths) > 0) {
  geometries <- do.call(c, lapply(valid_paths, st_geometry))
  combined_path <- st_union(st_sfc(geometries, crs = st_crs(dem_raster)))
  if (st_geometry_type(combined_path) == "MULTILINESTRING") {
    combined_path <- st_line_merge(combined_path)
  }
  combined_sf <- st_sf(geometry = combined_path, crs = st_crs(dem_raster))

  # Write to GPX
  st_write(combined_sf, "computed_paths_smoothed.gpx", driver = "GPX", delete_layer =
TRUE)

  cat("Pathfinding complete! Saved as GPX.\n")
} else {
  cat("No valid paths computed.\n")
}

# Print total distance
cat("Total distance of all computed paths:", round(total_distance, 2), "meters\n")

# Convert the DEM raster to a data frame for plotting
cat("Converting DEM to data frame...\n")
start_time_plotting <- Sys.time()  # Start time for plotting
dem_raster_df <- as.data.frame(dem_raster, xy = TRUE)  # Convert to data frame with x, y
coordinates
colnames(dem_raster_df)[3] <- "elevation"  # Rename the third column to 'elevation'

paths_sf <- do.call(st_sfc, lapply(valid_paths, st_geometry))

# Plot the DEM with the computed paths
cat("Plotting smoothed paths over DEM...\n")
plot <- ggplot() +
  geom_tile(data = dem_raster_df, aes(x = x, y = y, fill = elevation)) +  # Correct column
reference
  scale_fill_viridis_c() +
  geom_sf(data = paths_sf, color = "red", size = 1) +
  geom_sf(data = node_points, color = "blue", size = 2) +
  labs(title = "Smoothed Pathfinding over DEM", x = "Longitude", y = "Latitude") +
  theme_minimal()

# Print the plot to the screen
print(plot)
```



```
# Save the plot to a file (e.g., as PNG or PDF)
ggsave("smoothed_pathfinding_plot.png", plot = plot, width = 10, height = 8, dpi = 300)  #
Adjust file format and resolution

end_time_plotting <- Sys.time()
cat("Time taken for plotting the DEM with smoothed paths: ", difftime(end_time_plotting,
start_time_plotting, units = "secs"), "\n")

end_time_total <- Sys.time()
cat("Total time taken for all computations: ", difftime(end_time_total, start_time_total, units
= "mins"), "\n")

### END ###
```


Appendix 5: Pressure Drop, Pumping Station Placement Code

The following code takes input from the previously generated path files (in .GPX form) and performs the following functions:

1. Places tunnel between two defined points in one path (in this case, path_1.gpx which is the direct / optimal route between TGD and Baotou).
2. Performs pressure drop calculations to determine static head and frictional losses across each path.
3. Plots the elevation profile of each path indicating position of pumping stations.

```
gc()

# Load necessary libraries
library(sf)
library(ggplot2)
library(dplyr)
library(units)
library(xml2)
library(ggspatial)
library(elevatr)

# User-defined variables
flow_rate <- 2  # m^3/s
pipe_diameter <- 2 # meters
fluid_density <- 1000
fluid_viscosity <- 0.001
epsilon <- 0.00001 # meters (typical for GFRP pipe)
max_pump_head <- 160  # m

# Define tunnel ranges by route
# Route 1: Direct TGD to Baotou
tunnel_ranges <- list(
  list(start_km = 1417, end_km = 1446.6)  # for Route 1
)

calculate_friction_factor <- function(RE, epsilon, pipe_diameter) {
  if (RE < 2000) {
    return(64 / RE)
  } else {
    colebrook <- function(f) {
      return(1 / sqrt(f) + 2 * log10(epsilon / (3.7 * pipe_diameter) + 2.51 / (RE * sqrt(f))))
    }
    solution <- uniroot(colebrook, c(0.0001, 1))
    return(solution$root)
  }
}

estimate_pump_stations <- function(path_data, max_pump_head) {
  head_loss <- 0
  station_locations <- c()
  for (i in 2:nrow(path_data)) {
    if (!is.na(path_data$pressure_loss_friction[i]) &&
!is.na(path_data$pressure_loss_elevation[i])) {
      head_loss <- head_loss + (path_data$pressure_loss_friction[i] +
path_data$pressure_loss_elevation[i]) / (fluid_density * 9.81)
    }
    if (head_loss >= max_pump_head) {
      station_locations <- c(station_locations, path_data$cumulative_distance[i])
```



```
      head_loss <- 0
    }
  }
  return(station_locations)
}

compute_station_pressures <- function(path_data) {
  pressure_table <- data.frame()
  segment_loss <- 0
  for (i in 2:nrow(path_data)) {
    segment_loss <- segment_loss + path_data$pressure_loss_friction[i] +
path_data$pressure_loss_elevation[i]
    if (path_data$pump_stations[i] == 1) {
      upstream <- segment_loss
      downstream <- upstream + (fluid_density * 9.81 * max_pump_head)
      pressure_table <- rbind(pressure_table, data.frame(
        path = path_data$path[i],
        station_index = i,
        distance_km = path_data$cumulative_distance[i],
        upstream_pressure_MPa = upstream / 1e6,
        downstream_pressure_MPa = downstream / 1e6
      ))
      segment_loss <- 0
    }
  }
  return(pressure_table)
}

insert_tunnel <- function(df, start_km, end_km) {
  i1 <- which.min(abs(df$cumulative_distance - start_km))
  i2 <- which.min(abs(df$cumulative_distance - end_km))
  if (i2 > i1 + 1) {
    n_interp <- i2 - i1 - 1
    lon_seq <- seq(df$lon[i1], df$lon[i2], length.out = n_interp + 2)[-c(1, n_interp + 2)]
    lat_seq <- seq(df$lat[i1], df$lat[i2], length.out = n_interp + 2)[-c(1, n_interp + 2)]
    elev <- rep(min(df$elevation[i1], df$elevation[i2]), n_interp)  # flat tunnel
    tunnel_df <- data.frame(lon = lon_seq, lat = lat_seq, elevation = elev)
    df <- bind_rows(df[1:i1, ], tunnel_df, df[i2:nrow(df), ])
  }
  return(df)
}

process_gpx <- function(file_path, path_id, layer_name, apply_tunnel = FALSE,
tunnel_range = NULL) {
  gpx_data <- st_read(file_path, layer = layer_name, quiet = TRUE)
  if (nrow(gpx_data) == 0) stop(paste("No data in", file_path))

  gpx_points <- gpx_data %>%
    mutate(lon = st_coordinates(.)[,1], lat = st_coordinates(.)[,2]) %>%
    select(lon, lat)

  gpx_points <- gpx_points %>%
    mutate(prev_lon = lag(lon), prev_lat = lag(lat)) %>%
    rowwise() %>%
    mutate(distance = ifelse(is.na(prev_lon), 0,
                             geosphere::distHaversine(c(prev_lon, prev_lat), c(lon, lat)))) %>%
    ungroup() %>%
    mutate(cumulative_distance = cumsum(distance) / 1000)  # km
```



```
  elevations <- get_elev_point(st_as_sf(gpx_points, coords = c("lon", "lat"), crs = 4326),
                               prj = st_crs(4326)$proj4string, src = "aws")
  gpx_points$elevation <- elevations$elevation

  if (apply_tunnel && !is.null(tunnel_range)) {
    gpx_points <- insert_tunnel(gpx_points, tunnel_range$start_km, tunnel_range$end_km)
  }

  RE <- (4 * fluid_density * flow_rate) / (pi * fluid_viscosity * pipe_diameter)
  friction_factor <- calculate_friction_factor(RE, epsilon, pipe_diameter)

  gpx_points <- gpx_points %>%
    mutate(prev_elevation = lag(elevation, default = first(elevation)),
           elevation_change = elevation - prev_elevation,
           elevation_gain = ifelse(elevation_change > 0, elevation_change, 0),
           pipe_distance = sqrt(distance^2 + elevation_change^2),
           pressure_loss_friction = friction_factor * (pipe_distance / pipe_diameter) *
             (flow_rate^2 / (2 * 9.81 * (pipe_diameter / 2)^2)),
           pressure_loss_elevation = ifelse(elevation_change > 0, 9.81 * elevation_change *
fluid_density, 0),
           path = path_id)

  gpx_points$pump_stations <- 0
  pump_locations <- estimate_pump_stations(gpx_points, max_pump_head)
  gpx_points$pump_stations[gpx_points$cumulative_distance %in% pump_locations] <- 1

  return(gpx_points)
}

# Original
path1 <- process_gpx("path_1.gpx", "Direct TGD to Baotou", "route_points")
path2 <- process_gpx("path_2.gpx", "TGD to Baotou Via Xi'An", "track_points")

# With tunnels
path1_tunnel <- process_gpx("path_1.gpx", "TGD to Baotou (Tunneled)", "route_points",
TRUE, tunnel_ranges[[1]])

all_paths <- bind_rows(path1, path2, path1_tunnel)

# Compute pressure upstream/downstream of each pump station
pressure_table <- bind_rows(
  compute_station_pressures(path1),
  compute_station_pressures(path2),
  compute_station_pressures(path1_tunnel)
)

print(pressure_table)

# Plot elevation with overlaid paths and pumping stations
all_paths$path <- factor(all_paths$path, levels = c(
  "TGD to Baotou (Tunneled)",
  "Direct TGD to Baotou",
  "TGD to Baotou Via Xi'An"
))

plot <- ggplot(all_paths, aes(x = cumulative_distance, y = elevation, color = path, linetype =
path)) +
```



```
  geom_line(size = 0.5) +
  scale_linetype_manual(name = "Route", values = c(
    "TGD to Baotou (Tunneled)" = "dotted",
    "Direct TGD to Baotou" = "solid",
    "TGD to Baotou Via Xi'An" = "solid"
  )) +
  geom_point(data = all_paths %>% filter(pump_stations == 1 & !grepl("Tunneled", path)),
             aes(x = cumulative_distance, y = elevation, shape = "Booster Pumping Station"),
             color = "black", fill = "white", size = 2, stroke = 0.5) +
  scale_color_manual(values = c(
    "Direct TGD to Baotou" = "seagreen",
    "TGD to Baotou Via Xi'An" = "coral",
    "TGD to Baotou (Tunneled)" = "red"
  )) +
  scale_shape_manual(name = "", values = c("Booster Pumping Station" = 21)) +
  labs(title = "Elevation Profile with Booster Pumping Stations",
       x = "Distance Along Route (km)",
       y = "Elevation (m)",
       color = "Route") +
  theme_minimal() +
  theme(legend.position = c(0.05, 0.95),
        legend.justification = c("left", "top"),
        legend.box = "vertical",
        text = element_text(size = 14))

plot <- plot +
  geom_vline(data = tibble(
    km = c(tunnel_ranges[[1]]$start_km, tunnel_ranges[[1]]$end_km),
    label = rep("Tunnel Entry/Exit", 2)
  ),
  aes(xintercept = km), linetype = "dashed", color = "gray40", linewidth = 0.3) +
  geom_text(data = tibble(
    km = c(tunnel_ranges[[1]]$start_km, tunnel_ranges[[1]]$end_km),
    elevation = rep(Inf, 2),
    label = c("Tunnel In", "Tunnel Out")
  ),
  aes(x = km, y = elevation, label = label),
  inherit.aes = FALSE,
  vjust = -0.2, size = 2.5, color = "gray30")

print(plot)

# Compute required totals per route with units
total_summary <- all_paths %>%
  group_by(path) %>%
  summarise(
    total_distance_km = set_units(sum(distance, na.rm = TRUE) / 1000, "km"),
    total_elevation_gain = set_units(max(elevation, na.rm = TRUE) - min(elevation, na.rm =
TRUE), "m"),
    cumulative_elevation_gain = set_units(sum(ifelse(elevation_change > 0, elevation_change,
0), na.rm = TRUE), "m"),
    total_static_pressure_drop = set_units(9.81 * (max(elevation, na.rm = TRUE) -
min(elevation, na.rm = TRUE)) * fluid_density, "Pa"),
    total_frictional_pressure_drop = set_units(sum(pressure_loss_friction, na.rm = TRUE),
"Pa"),
    number_of_pump_stations = sum(pump_stations, na.rm = TRUE),
    .groups = "drop"
  )
```



```
total_summary <- total_summary %>%
  mutate(
    total_head_m = (total_static_pressure_drop + total_frictional_pressure_drop) /
(fluid_density * 9.81),
    total_hydraulic_power_W = fluid_density * 9.81 * flow_rate * set_units(total_head_m,
NULL),
    total_pump_power_W = total_hydraulic_power_W / 0.828,
    total_pump_power_MW = set_units(total_pump_power_W, "MW")
  )

print(total_summary)

### END ###
```

Appendix 6: Matthew Code (rename)

Earthquake frequency code

```
    # gc()
    #
    #
    # # Load necessary libraries
    # library(tidyverse)
    # library(rpart)
    # library(geosphere)
    # library(elevatr)
    # library(sf)
    # library(ggmap)
    # library(ggplot2)
    # library(ggspatial)
    # library(dplyr)
    # library(units)
    # library(lpSolve)
    # library(caret)
    #
    #
    # # Load your TSV file (earthquake history data)
    # earthquake_data <- read.delim("seismic_data.tsv", sep = "\t", header = TRUE)
    # earthquake_data <- earthquake_data %>%
    #   rename(lat = Latitude, lon = Longitude, magnitude = Mag)
    #
    #
    # # Remove rows with NA values in magnitude and depth
    # earthquake_data_filter <- earthquake_data %>%
    #   filter(((lat > 39) & (lat < 41)) & ((lon > 112) & (lon < 115)))

    gc()

    # Load necessary libraries
    library(dplyr)
    library(ggplot2)

    # Function to read and filter earthquake data
    filter_earthquake_data <- function(file_path, min_lat, max_lat, min_long, max_long) {
      # Read the TSV file
      earthquake_data <- read.delim(file_path, header = TRUE, sep = "\t")
```



```
  # Filter data based on latitude and longitude range
  filtered_data <- earthquake_data %>%
    filter(Latitude >= min_lat & Latitude <= max_lat & Longitude >= min_long & Longitude <= max_long)

  return(filtered_data)
}

# Function to perform Gutenberg-Richter analysis
gutenberg_richter_analysis <- function(filtered_data) {
  # Calculate the cumulative number of earthquakes for each magnitude
  magnitude_counts <- filtered_data %>%
    group_by(Mag) %>%
    summarise(count = n()) %>%
    arrange(desc(Mag)) %>%
    mutate(cumulative_count = cumsum(count))

  # Perform linear regression on log10(cumulative_count) vs. Mag
  magnitude_counts <- magnitude_counts %>%
    mutate(log_cumulative_count = log10(cumulative_count))

  # Filter out non-finite values
  magnitude_counts <- magnitude_counts %>%
    filter(is.finite(log_cumulative_count))

  regression_model <- lm(log_cumulative_count ~ Mag, data = magnitude_counts)

  # Extract coefficients
  a <- coef(regression_model)[1]
  b <- -coef(regression_model)[2]

  return(list(a = a, b = b, model = regression_model, data = magnitude_counts))
}

# Function to calculate the probability of an earthquake of magnitude >= M
calculate_probability <- function(a, b, M) {
  # Calculate the number of earthquakes of magnitude M or greater
  log_N = a - b * M
  N = 10^log_N

  # Calculate the total number of earthquakes
  total_earthquakes = 10^a

  # Calculate the probability
  probability = N / total_earthquakes

  return(probability)
}

# Function to calculate the average number of years before an earthquake of magnitude >= M
calculate_average_years <- function(a, b, M) {
  # Calculate the number of earthquakes of magnitude M or greater
  log_N = a - b * M
  N = 10^log_N

  # Calculate the total number of earthquakes per year
  total_earthquakes_per_year = 10^a
```

```
  # Calculate the probability
  probability = N / total_earthquakes_per_year

  # Calculate the average number of years before an earthquake of magnitude M or greater
  average_years = 1 / probability

  return(average_years)
}

# Main function
main <- function() {
  # Define file path and latitude/longitude range
  file_path <- "seismic_data.tsv"
  min_lat <- 38
  max_lat <- 40
  min_long <- 110
  max_long <- 115

  # Filter earthquake data
  filtered_data <- filter_earthquake_data(file_path, min_lat, max_lat, min_long, max_long)
  print(filtered_data)
  # Perform Gutenberg-Richter analysis
  analysis_results <- gutenberg_richter_analysis(filtered_data)

  # Print results
  cat("Gutenberg-Richter coefficients:\n")
  cat("a =", analysis_results$a, "\n")
  cat("b =", analysis_results$b, "\n")

  # Calculate and print the probability of an earthquake of magnitude 4 or higher
  probability <- calculate_probability(analysis_results$a, analysis_results$b, 4)
  cat("Probability of an earthquake of magnitude 4 or higher:", probability, "\n")

  # Calculate and print the average number of years before an earthquake of magnitude 4 or higher
  average_years <- calculate_average_years(analysis_results$a, analysis_results$b, 5)
  cat("Average number of years before an earthquake of magnitude 4 or higher:", average_years, "\n")

  # Plot the results
  ggplot(analysis_results$data, aes(x = Mag, y = log_cumulative_count)) +
    geom_point() +
    geom_smooth(method = "lm", se = FALSE, color = "blue") +
    labs(title = "Gutenberg-Richter Relationship",
         x = "Magnitude",
         y = "Log10(Cumulative Count)") +
    theme_minimal()
}

# Run the main function
main()
```

Pipe failure code

```
# Load necessary library
library(dplyr)

# Define a function to calculate the reliability function
reliability_function <- function(lambda, t) {
  exp(-lambda * t)
```



```
}

# Define a function to calculate the cumulative distribution function (CDF)
cdf_function <- function(lambda, t) {
  1 - exp(-lambda * t)
}

# Define a function to calculate the maintenance schedule
maintenance_schedule <- function(pipeline_sections, time_period) {
  pipeline_sections %>%
    rowwise() %>%
    mutate(
      AdjustedFailureRate = FailureRate * Length,  # Adjust failure rate by length
      Reliability = reliability_function(AdjustedFailureRate, time_period),
      CDF = cdf_function(AdjustedFailureRate, time_period),
      MaintenanceNeeded = ifelse(CDF > 0.5, "Yes", "No")
    )
}

# Define a function to calculate the optimal maintenance interval
optimal_maintenance_interval <- function(pipeline_sections, target_reliability) {
  cumulative_failure_rate <- sum(pipeline_sections$FailureRate * pipeline_sections$Length)
  interval <- -log(target_reliability) / cumulative_failure_rate
  return(interval)
}

# Example pipeline sections with different materials, failure rates, and lengths
pipeline_sections <- data.frame(
  Section = c("Stainless Steel", "Polyethylene", "GFRP"),
  FailureRate = c(0.01, 0.02, 00.00029),  # Failure rates per year per unit length
  Length = c(0, 0, 1926)  # Lengths of each section in kilometers
)

# Define the target reliability (e.g., 0.95 for 95% reliability)
target_reliability <- 0.95

# Calculate the optimal maintenance interval
maintenance_interval <- optimal_maintenance_interval(pipeline_sections, target_reliability)

# Print the maintenance interval
cat("Optimal Maintenance Interval (years):", maintenance_interval, "\n")
cat("Optimal Maintenance Interval (months):", maintenance_interval*12, "\n")

# Calculate the maintenance schedule for the given time period
time_period <- maintenance_interval
maintenance_schedule(pipeline_sections, time_period)
```

Appendix 7: Costing/Energy

Costing was calculated in excel using various assumptions/calculations

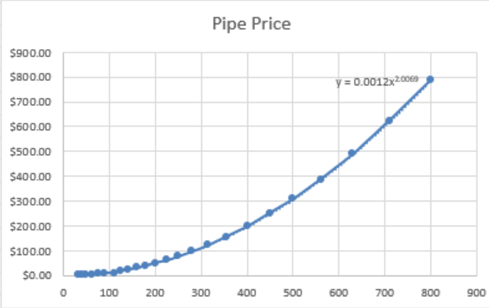
Pipeline Material Cost:

*All costs are per meter

Size (mm)	PE100 PN4 SDR41	PE100 PN6.3 SDR26	PE100 PN8 SDR21	PE100 PN10 SDR17	PE100 PN12.5 SDR13.6	PE100 PN16 SDR11
32	-	-	$1.11	$1.30	$1.65	$1.89
40	-	-	$1.66	$2.05	$2.49	$3.01
50	-	-	$2.63	$3.16	$3.84	$4.67
63	-	$3.32	$4.02	$5.04	$6.12	$7.38
75	-	$4.70	$5.75	$7.11	$8.54	$10.31
90	-	$6.83	$8.32	$8.56	$12.27	$14.94
110	$6.55	$10.23	$12.47	$10.23	$18.41	$22.15
125	$8.60	$12.91	$15.94	$19.42	$23.73	$28.74
140	$10.78	$16.29	$19.89	$24.37	$29.71	$35.82
160	$13.97	$21.37	$26.13	$31.67	$38.78	$47.02
180	$17.39	$26.61	$32.85	$40.25	$49.23	$59.43
200	$21.38	$33.00	$40.68	$49.65	$60.36	$73.28
225	$27.06	$41.47	$51.42	$62.98	$76.66	$92.76
250	$33.90	$51.38	$62.92	$77.19	$94.44	$114.09
280	$42.02	$64.09	$79.42	$96.97	$118.33	$143.03
315	$52.78	$81.64	$99.79	$122.80	$149.97	$181.05
355	$67.14	$103.16	$126.68	$156.27	$190.11	$229.79
400	$85.10	$130.83	$161.54	$197.50	$241.10	$291.76
450	$107.32	$165.43	$204.30	$250.24	$305.41	$369.47
500	$133.60	$204.06	$252.04	$308.25	$376.83	$455.87
560	$166.35	$255.76	$315.45	$387.30	$472.73	$571.06
630	$210.47	$324.10	$398.48	$489.41	$597.57	$723.60
710	$267.83	$411.97	$507.41	$622.44	$759.16	-
800	$339.62	$521.76	$643.10	$789.27	$962.97	-

https://www.matrixpiping.com.au/pages/poly-pipe-prices

Size	Price
32	$1.30
40	$2.05
50	$3.16
63	$5.04
75	$7.11
90	$8.56
110	$10.23
125	$19.42
140	$24.37
160	$31.67
180	$40.25
200	$49.65
225	$62.98
250	$77.19
280	$96.97
315	$122.80
355	$156.27
400	$197.50
450	$250.24
500	$308.25
560	$387.30
630	$489.41
710	$622.44
800	$789.27



Extrapolated pipe	755	* In Australia
	529	*In China
$/km/1m dia	528,604	China Adjusted
$/km/1m dia	755,149	Unadjusted

3. **Plastics & Composites**
   - China produces a large volume of polymers and composite materials.
   - Australia imports many plastic raw materials, making costs **20-40% higher.**

Battery costs

	Option A	Option B	Option C		Notes
Number of stations	22	23	34		
Total energy req (MWh p.a)	289080	359160	420480		*MWh/year * annual energy requirement
Load per station (MW)	1.50	1.78	1.41		* With annual load of xMW, finding amount per station by dividing by number of stations and assuming equal load across all stations for calculation purposes
Backup supply (MWh)	12	14.26	11.29		*8-hour backup supply
Total Battery Capacity (MWh)	264	328	384		*backup supply * # of stations
Cost	23,232,000	28,864,000	33,792,000		* cost of battery $88 USD/kWh

CAPEX

Category	Option A: Tunneling (USD)	Option B: Pumping (USD)	Option C: Indirect (USD)	Justification
Land Acquisition	142,349,288	144,449,278	193,724,031	USD $2,500/mu (mu=666.67 m2), 20m width
Pipeline Materials	2,163,720,000	2,195,640,000	2,944,620,000	Pipeline cost @ USD $760k/km, based on total route distance, scaling factor of 1.5 to account for monitoring systems, fittings etc
Pumping Infrastructure	506,000,000	552,000,000	1,207,000,000	Based on word table in section 3
Water Intake System	1,000,000,000	1,000,000,000	1,000,000,000	
Construction Costs	2,442,726,000	2,478,762,000	3,324,321,000	$33,000/in/km (cpi adjusted from 1996)
Tunneling Costs (Option A)	6,000,000,000	-	-	Tunnelling cost estimate by matt
Pumping Over Mountain Costs (Option B)	-	1,000,000,000	1,500,000,000	Additional construction for elevation and terrain access
Battery Cost	23,232,000	28,864,000	33,792,000	Calculated on Battery Reqs sheet
Contingency (10%)	1,225,479,529	737,085,128	1,016,966,503	10% contingency for unforeseen CAPEX elements
Total	13,518,506,817	8,151,800,406	11,235,423,535	added 15mil from Treatment cost

Tunnel Option OPEX

Year	Energy Cost (USD)	Maintenance (USD)	Labor (USD)	Admin & Insurance (USD)	Environmental Monitoring (USD)	Total OPEX (USD)	Assumptions
2025	13,490,400	11,190,000	43,274,400	13,518,507	4,476,000	85,949,307	* assuming operation all year round, 8760MWh annually
2026	13,895,112	11,525,700	44,572,632	13,924,062	4,610,280	88,527,786	*Hydro = 42 USD /MWh
2027	14,311,965	11,871,471	45,909,811	14,341,784	4,748,588	91,183,620	*assuming 1/3rd hydro 2/3 solar
2028	14,741,324	12,227,615	47,287,105	14,772,037	4,891,046	93,919,128	*Solar = 49 USD/MWh
2029	15,183,564	12,594,444	48,705,718	15,215,199	5,037,777	96,736,702	* energy cost = (((1/3)*42*8760*PUMP REQ)+((2/3)*49*8760*PUMP REQ))
2030	15,639,071	12,972,277	50,166,890	15,671,654	5,188,910	99,638,803	* Assuming 3% cost increases per year
2031	16,108,243	13,361,445	51,671,897	16,141,804	5,344,578	102,627,967	*Assuming pump efficiency 0.75
2032	16,591,490	13,762,289	53,222,054	16,626,058	5,504,915	105,706,806	*assume $5000/km maintenance annually
2033	17,089,235	14,175,157	54,818,715	17,124,840	5,670,062	108,878,010	*Assume labour costs = 2% pipeline capex + altitude construction
2034	17,601,912	14,600,412	56,463,277	17,638,585	5,840,164	112,144,351	*Assume environmental monitoring = 40% maintenance
2035	18,129,970	15,038,424	58,157,175	18,167,743	6,015,369	115,508,681	*assume insurance rate of 5% capex split over 50 years with 3% interest p.a
2036	18,673,869	15,489,577	59,901,890	18,712,775	6,195,830	118,973,942	*added 1.7mil to maintenance from karinnas opex
....	....	....	....	....	....	....	....

Direct Mountain Option OPEX

Year	Energy Cost (USD)	Maintenance (USD)	Labor (USD)	Admin & Insurance (USD)	Environmental Monitoring (USD)	Total OPEX (USD)	Assumptions
2025	16,760,800	11,330,000	63,912,800	8,151,800	4,532,000	104,687,400	* assuming operation all year round, 8760MWh annually
2026	17,263,624	11,669,900	65,830,184	8,396,354	4,667,960	107,828,022	*Hydro = 42 USD /MWh
2027	17,781,533	12,019,997	67,805,090	8,648,245	4,807,999	111,062,863	*assuming 1/3rd hydro 2/3 solar
2028	18,314,979	12,380,597	69,839,242	8,907,692	4,952,239	114,394,749	*Solar = 49 USD/MWh
2029	18,864,428	12,752,015	71,934,419	9,174,923	5,100,806	117,826,591	* energy cost = (((1/3)*42*8760*PUMP REQ)+((2/3)*49*8760*PUMP REQ))
2030	19,430,361	13,134,575	74,092,452	9,450,171	5,253,830	121,361,389	* Assuming 3% cost increases per year
2031	20,013,272	13,528,613	76,315,226	9,733,676	5,411,445	125,002,231	*Assuming pump efficiency 0.75
2032	20,613,670	13,934,471	78,604,682	10,025,686	5,573,788	128,752,298	*assume $5000/km maintenance annually
2033	21,232,080	14,352,505	80,962,823	10,326,457	5,741,002	132,614,867	*Assume labour costs = 2% pipeline capex + altitude construction
2034	21,869,042	14,783,080	83,391,708	10,636,251	5,913,232	136,593,313	*Assume environmental monitoring = 40% maintenance
2035	22,525,114	15,226,573	85,893,459	10,955,338	6,090,629	140,691,112	*assume insurance rate of 5% capex split over 50 years with 3% interest p.a
2036	23,200,867	15,683,370	88,470,263	11,283,998	6,273,348	144,911,845	*added 1.7mil to maintenance from karinnas opex
….	….	….	….	….	….	….	….

Indirect Route OPEX

Year	Energy Cost (USD)	Maintenance (USD)	Labor (USD)	Admin & Insurance (USD)	Environmental Monitoring (USD)	Total OPEX (USD)	Assumptions
2025	19,622,400	14,615,000	88,892,400	11,235,424	5,846,000.0	140,211,224	* assuming operation all year round, 8760MWh annually
2026	20,211,072	15,053,450	91,559,172	11,572,486	6,021,380.0	144,417,560	*Hydro = 42 USD /MWh
2027	20,817,404	15,505,054	94,305,947	11,919,661	6,202,021.4	148,750,087	*assuming 1/3rd hydro 2/3 solar
2028	21,441,926	15,970,205	97,135,126	12,277,251	6,388,082.0	153,212,590	*Solar = 49 USD/MWh
2029	22,085,184	16,449,311	100,049,179	12,645,568	6,579,724.5	157,808,967	* energy cost = (((1/3)*42*8760*PUMP REQ)+((2/3)*49*8760*PUMP REQ))
2030	22,747,740	16,942,791	103,050,655	13,024,935	6,777,116.2	162,543,236	* Assuming 3% cost increases per year
2031	23,430,172	17,451,074	106,142,174	13,415,683	6,980,429.7	167,419,533	*Assuming pump efficiency 0.75
2032	24,133,077	17,974,607	109,326,440	13,818,154	7,189,842.6	172,442,119	*assume $5000/km maintenance annually
2033	24,857,069	18,513,845	112,606,233	14,232,698	7,405,537.9	177,615,383	*Assume labour costs = 2% pipeline capex + altitude construction
2034	25,602,781	19,069,260	115,984,420	14,659,679	7,627,704.0	182,943,845	*Assume environmental monitoring = 40% maintenance
2035	26,370,865	19,641,338	119,463,952	15,099,470	7,856,535.2	188,432,160	*assume insurance rate of 5% capex split over 50 years with 3% interest p.a
2036	27,161,991	20,230,578	123,047,871	15,552,454	8,092,231.2	194,085,125	*added 1.7mil to maintenance from karinnas opex
….	….	….	….	….	….	….	….